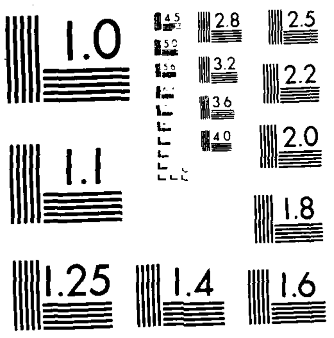
AD-A195 932 DYNAMICAL SYSTEMS AND MOTION VISION(U) MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB J HEEL APR 88 AI-M-1037 N00014-85-K-0124 1/1

UNCLASSIFIED F/G 12/9 NL


1.0
4.5
5.0
5.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DTIC FILE COPY

4

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AI Memo 1037	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Dynamical Systems and Motion Vision		5. TYPE OF REPORT & PERIOD COVERED: memorandum
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Joachim Heel		8. CONTRACT OR GRANT NUMBER(s): N00014-85-K-0124
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Artificial Inteligence Laboratory, 545 Technology Square, Cambridge, MA 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Advanced Research Projects Agency, 1400 Wilson Blvd., Arlington, VA 22209		12. REPORT DATE: April 1988
		13. NUMBER OF PAGES: 54
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Office of Naval Research, Information Systems, Arlington, VA 22217		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report): Distribution is unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES: None		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number): dynamical systems, motion vision, Kalman filter, depth maps, motion recovery		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)		

DTIC ELECTE JUL 27 1988 S D
D

AD-A195 932

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I.Memo No. 1037 April, 1988

Dynamical Systems and Motion Vision

Joachim Heel

Abstract: In this paper ~~we~~ show how the theory of dynamical systems can be employed to solve problems in motion vision. In particular we develop algorithms for recovery of dense depth maps and motion parameters using state space observers or filters. ~~We begin~~ with a review of previous, related work followed by an overview of relevant aspects of the theory of dynamical systems. Three dynamical models of the imaging situation are presented:

- ~~In~~ the first model ~~we~~ assume that motion is known and a reflectance model for the surface is given. Depth is recovered directly from brightness measurements with a nonlinear Kalman filter.
- The second model assumes that a planar surface is being viewed. Motion and depth are recovered with a nonlinear observer.
- The third model makes no explicit assumptions about motion or surface. It is embedded in a system that iteratively improves an estimate for both the motion parameters and a dense depth map using Kalman filters.

No feature-matching preprocessor is required. Solutions to other motion vision problems (tracking, camera parameter estimation) using dynamical systems theory are suggested.

Acknowledgements: This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the Laboratory's Artificial Intelligence Research is provided in part by the Advanced Research Projects Agency under Office of Naval Research contract N00014-85-K-0124.

1 Introduction

Research in motion vision has previously focussed on the interpretation of two successive image frames in which either the camera or parts of the environment are in motion. Consequently, the results of these methods are instantaneous and make no use of the redundancy inherent in a series of frames. Examples of such *instantaneous* analyses include: estimation of the optical flow (Hildreth [19], [20], Horn and Schunck [27], Nagel and Enkelmann [40], [39], [41], Anandan [2], Fennema and Thompson [12]), recovery of motion parameters (Longuet-Higgins and Prazdny [35], Negahdaripour, Weldon and Horn [42], [28], Weng, Huang and Ahuja [64], Fennema and Thompson [12], Tsai and Huang [55], [56], Waxman and Wohn [63], Roach and Aggarwal [47], Prazdny [45], Waxman and Ullman [62]) and depth measurement (Kanatani [31], Prazdny [45], Waxman, Sinha and Ullman [62], [61], and Mitiche [38]).

Recently, some proposals have been made for making use of more than two frames out of an image sequence. Additional frames contain redundant information that may provide additional constraints on an otherwise *ill-posed problem*.

Several categories of multiframe motion algorithms have evolved: A first distinction can be made with respect to the format of the input data.

- *Feature-based methods.* These methods assume that the localization and correspondence of features has been established beforehand by some other module. Since features are usually sparse depth cannot be recovered densely.

- *Non-feature-based methods.* In most of these algorithms a brightness constancy assumption similar to the one first derived by Horn and Schunck [27] is made to determine how the projections of real world points move in the image plane.

The second distinction deals with the input data organization. The approaches taken here are

- *Recursive algorithms.* These methods commonly maintain some current estimate, accept one frame of input data at a time and continuously update the estimate.

- *Global (batch) algorithms.* Here, all of the input data is accepted at once and some global criterion is optimized as opposed to the local improvement achieved by the recursive schemes. The main disadvantage, however, is the necessity of storing large amounts of data and not being suitable for "on-line" applications.

One example of a structure-from-motion algorithm that exploits information in a sequence of frames is Ullman's incremental rigidity scheme [58]. In this scheme a set of feature points is tracked through a sequence of images. Beginning with an initially flat model the algorithm incorporates the incoming information about the displacement of the features and updates the model while maximizing its rigidity. The incremental rigidity scheme is a recursive feature-based method and therefore suitable for on-line recovery of depth information in sparse locations.

Another feature-based algorithm is the one presented by Sethi and Jain [49]. They assume that a number of features has been localized in each frame of an image sequence.

They then show how the smoothness of motion can be employed to find correspondences among these features i.e. they propose a solution to the correspondence problem over a set of motion frames. Using a greedy optimization algorithm, they address the following problem: given m points in each of n frames, determine m trajectories. The solution is found by minimizing a functional capturing a general smoothness constraint for the trajectory path.

Tsai [54] presents a surface reconstruction procedure for known viewing orientations extended to n viewing positions. He shows that the multiframe method provides higher accuracy while only slightly increasing the computational complexity. This will be the decisive efficiency measure for multiframe algorithms: can the accuracy be increased sufficiently with respect to the additional computational cost.

Tracking is another application of multiframe motion vision. Usually the idea is to control the motion of a camera viewing a certain moving object such that the object image remains nearly fixed in the image plane. This application is mostly feature-based and an example of a purely recursive scheme. There are numerous publications on this subject. Extensive reference lists are provided by [10] and [3].

Shariat and Price [50] use point correspondences in more than two frames to estimate the motion of a moving object. They exploit the fact that after compensating for translation, points on the observed object will move in circles in space according to the rotational velocity. The motion estimation problem can then be reduced to the problem of computing circle parameters from point correspondences. This approach can be classified as a feature-based global solution since measurements from all frames must be available before the parameters can be computed.

Another recursive depth estimation procedure for more than two frames was suggested by Bharwani, Riseman and Hanson [5]. A correlation based method similar to the one employed in section 6 is used to estimate displacements of features between successive frames. Under the assumption of known translational motion the depth is then easily recovered. This recursive approach is not really restricted to identifiable features and pure translation but it becomes much easier to handle mathematically. An important feature of the algorithm is that it exploits the fact that motion is known to predict the displacement in the next frame and thereby limit the search in the correlation-based estimator. However, a more physically motivated strategy for predicting and updating the estimates that involves the minimization of measurement errors seems necessary. In addition, we would like to avoid restricting the motion to known translation.

The above methods employ different approaches for dealing with the data sampled over time, which are largely motivated by physically intuitive constraints imposed upon the problem (for instance rigidity of the object, smoothness of the trajectories). Researchers have been looking for a more systematic way in which to incorporate the temporal relationship among the successive measurements. Recently problems in motion vision have been formulated in a way that deals explicitly with the time-dependency of the observed data. Such a formulation is provided by the theory of dynamical systems.

One of the earliest applications of dynamical systems methodology may be found in Stuller and Krishnamurthy [52]. As we will see, a Kalman filter is a means of estimating states of a dynamical system when only a function of these states is measurable as output. Stuller and Krishnamurthy formulate the projected translational motion of an object in terms of dynamical systems equations in which the interframe displacements appear as states so that a Kalman filter can be employed to estimate them. The authors present experimental applications of their method. Apart from the restriction to translational motion the approach has a severe flaw: the underlying dynamical equations constitute a merely heuristic association of the variables needed for the estimation. When mathematical models are not based on laws of physics their relationship to reality becomes unclear.

Besides investigating the relationship between the motion-field and the optical flow, Verri and Poggio [60] noted that the motion field, the projection of the 3D velocity field into the image plane, is a vector field tangential to the trajectories of a dynamical system and that the qualitative properties of this dynamical field are reflected in the properties of the optical flow field that is commonly recovered from brightness variations. They give an overview of the theory of dynamical systems from the theoretical perspective of [21].

Verri, Girosi and Torre [59] establish a formal correspondence between the theory of dynamical systems and motion vision. Their basic observation is that the motion field equations form a two-dimensional dynamical system. Properties of this dynamical system can be inferred from its behavior near singular points that may appear in images as a focus of expansion, for instance. They show how the type of motion as well as quantitative measurements of rotation and translation may be determined given the optical flow near the singular points.

Direct applications of the theory of dynamical systems to the estimation of object kinematics and structure in motion vision have been discussed repeatedly by Broida and Chellappa. In [6] the authors present a dynamical system that describes the temporal variation of 3D coordinates and velocity. They further present a model of the measurement process corrupted by noise. Given such a formulation, the technique of optimal filtering is applied to estimate depth and velocity. The scheme suffers from an extremely simplified description of object motion (only two real-world points with known position relative to the center of mass are described by the model), the motion is planar (translation and rotation in a plane only) and the measurement equations assume known interframe correspondences (no relationship to the measurable brightness values is established).

In [7] the model is generalized to n feature points but a known matching between the frames is still assumed. Again the iterated extended Kalman filter is employed to perform the recursive estimation. Unit quaternions are used to represent interframe rotation. No application of the technique to real data is mentioned, presumably because the preconditions are yet too restrictive. The authors have also proposed a corresponding global solution [8] which again shows the duality of the two approaches.

Interesting practical results are presented by Dickmans [11]. The objective in this paper is to recover real-world coordinates and motion parameters from the measured image point coordinates of distinguished feature points. This requires assumptions about the underlying dynamics of the object being viewed to obtain a complete dynamical systems description, which is then used in a Kalman filter estimation procedure. Although the paper does not present the exact form of the models employed, extensive experiments involving docking procedures for a vehicle, landing of aircraft and guidance of motor vehicles are reported.

Recently Matthies, Szeliski and Kanade [37] have presented ideas very similar to our own. They propose a non-feature-based recursive dense depth estimation based on Kalman filtering. Motion is assumed to be known, purely translational and parallel to the image plane. A correlation estimator is used to compute displacements of image points between successive frames. If motion is known, depth can be recovered in a straightforward manner. The key idea is to use the Kalman filter to update a dense depth map given these correlation-based measurements for every new frame. Although severely restricted by the necessity to know motion, encouraging experimental results are presented.

The main intention of this paper is to introduce a rigorous dynamical systems framework for the motion vision process. This begins by realizing that there is no one dynamical system that describes the behavior of successive frame measurements "correctly". The structure of the system depends crucially on the assumptions made about the imaging situation, which are mainly reflected in the output equations. We describe three different dynamical models of motion vision appropriate for various imaging situations. As an example of how techniques used in the control of dynamical systems may be employed in motion vision we develop recursive algorithms which do not require the existence of feature-correspondence for the estimation of dense depth maps and motion parameters using Kalman filters. We stress the importance of a rigorous physical motivation of the model of the imaging process, which is a major flaw of many previous approaches.

We proceed as follows. The subsequent section introduces the terminology and notation of dynamical systems. We then describe our choice of coordinate systems and derive the basic motion equations for both the continuous and the discrete case. Then three dynamical models of the imaging situation are presented and corresponding estimators for depth and motion are derived.

2 Dynamical Systems

This section contains an overview of the relevant material from the theory of dynamical systems. The reader familiar with the topic may wish to skip this section. Literature in this area is readily available. [36], [14] and [30] provide introduction to the general theory dealing mainly with continuous, linear systems. Discrete systems - of particular importance in our case of a sampling camera system - are discussed in [32], [44], [65] and [13]. Special topics such as identification may be found in [43] and [34]; for filtering a widely used reference (cf. [52], [6], [7]) is the one by Gelb [16].

2.1 Basic Concepts

We present here the *state space* model of dynamical systems. In the most general case such a system is described by a first-order vector differential equation

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$

and an algebraic vector equation

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)). \tag{2}$$

Equation (1) is referred to as the *plant* equation that describes the dynamic behavior of the system. It states that the temporal variation of the *state vector* $\mathbf{x}(t)$ depends (possibly in a nonlinear fashion) on $\mathbf{x}(t)$ itself and an *input vector* $\mathbf{u}(t)$. The *output equation* (2) models the measurement process in which the *output vector* $\mathbf{y}(t)$ is obtained as a function of the current state and the input.

Linear systems are a subset of the above, the model being

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) && (3) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t). && (4) \end{aligned}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are matrices. Most available methods operate on linear systems as we will see. We often visualize dynamical systems using block diagrams as in figure (1).

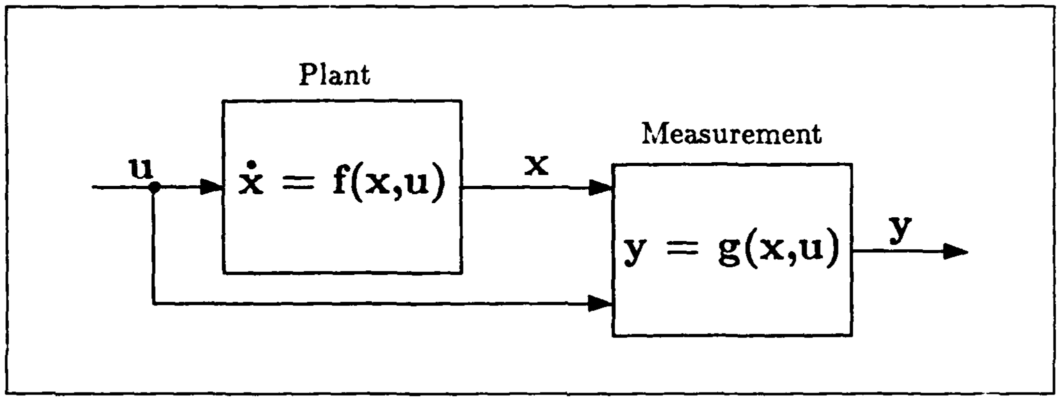


Figure 1: Block diagram representation of dynamical systems

The major task in the application of dynamical systems theory is constructing an appropriate model. We thereby encounter a tradeoff between the model's complexity and the accuracy with which it represents the actual dynamics.

2.2 An Example

Consider a mass m attached to a spring of stiffness k. The attachment of the spring may be moved by external influence by an amount $u(t)$. Finally we let $x(t)$ denote distance between the mass and the attachment. The arrangement is depicted in figure (2).

Summing up the forces on mass m provides the equation

$$m\ddot{x} = mg - k(x - u). \tag{5}$$

By introducing $\hat{x} = x - mg/k$, which simply describes the distance from the equilibrium $x_o = mg/k$, we obtain

$$\ddot{\hat{x}} = -\frac{k}{m}\hat{x} + \frac{k}{m}u. \tag{6}$$

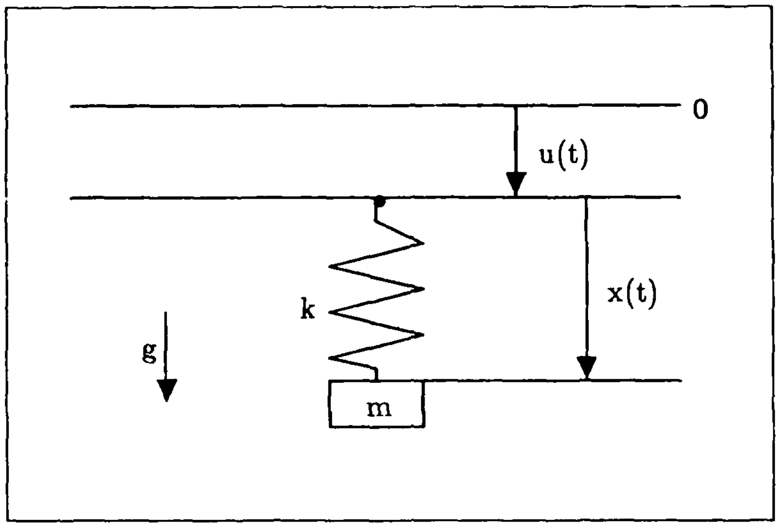


Figure 2: A simple dynamical system

Now we use the dummy variable $v = \dot{x}$ (simply the velocity) to convert the second-order ODE into two first order ODE's which are the desired plant equations:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{k}{m} \end{bmatrix} u. \tag{7}$$

We see that the plant equations are linear corresponding to (3). We can measure the behavior of this system by the position of the mass, which is simply the first component of the state vector. So the output equation is

$$y = \hat{x} = [1,0]\mathbf{x}. \tag{8}$$

2.3 Discrete Systems

For many systems, output and state are only defined at discrete points in time. We may also think of such systems as the result of a continuous system being sampled periodically. Assuming system (3), (4) being subject to sample and hold of period T, we can write

$$\mathbf{x}_{k+1} = \mathbf{\Phi}\mathbf{x}_k + \mathbf{\Gamma}\mathbf{u}_k \tag{9}$$
$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \mathbf{D}\mathbf{u}_k \tag{10}$$

where $\mathbf{x}_k$ is a shorthand for $\mathbf{x}(Tk)$, $k = 0, 1, \ldots$ and

$$\mathbf{\Phi} = e^{\mathbf{A}T} = \sum_{k=0}^{\infty} \frac{1}{k!}\mathbf{A}T \tag{11}$$
$$\mathbf{\Gamma} = (\int_0^T e^{\mathbf{A}\tau} d\tau)\mathbf{B}. \tag{12}$$

This description is suitable for dynamical systems involving digital computers, especially for processing of image frames taken at discrete points of time.

2.4 Linearization

Unfortunately, many realistic problems do not lead to such simple linear differential equations but appear in the more general form of (1), (2). Very few methods for the tasks discussed in the following sections are available, which operate directly on the nonlinear description. A remedy is to linearize the system by using a Taylor series expansion about some fixed point $\mathbf{x}_0$, $\mathbf{u}_0$:

$$\mathbf{f}(\mathbf{x},\mathbf{u}) = \mathbf{f}(\mathbf{x}_0,\mathbf{u}_0) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_0}(\mathbf{x}-\mathbf{x}_0) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{u}_0}(\mathbf{u}-\mathbf{u}_0) + \cdots \qquad (13)$$

If we assume that state and input vary only slightly around the point about which $\mathbf{f}$ has been expanded, we may neglect higher-order terms and write

$$\Delta\dot{\mathbf{x}} = \mathbf{A}\Delta\mathbf{x} + \mathbf{B}\Delta\mathbf{u} \qquad (14)$$

where we introduced $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0$, $\Delta\mathbf{u} = \mathbf{u} - \mathbf{u}_0$ and

$$\mathbf{A} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_0} \qquad \text{and} \qquad \mathbf{B} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{u}_0} \qquad (15)$$

are Jacobians of $\mathbf{f}$. A similar argument leads to a linearized output equation and we may apply linear techniques to the transformed system.

2.5 Stability

We differentiate two definitions of stability (see [15], [30], [36]):
External (BIBO) stability: A dynamical system as in (1), (2) is said to be externally stable if a bounded input function $\mathbf{u}(t)$ results in a bounded output function $\mathbf{y}(t)$.
Internal (asymptotic) stability: A *linear* dynamical system as in (3), (4) is said to be internally stable if the solution of

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) \qquad (16)$$

tends toward zero as $t \rightarrow \infty$ for arbitrary initial condition $\mathbf{x}(t_0)$.
A necessary and sufficient condition for internal stability (also called asymptotic stability) is that all eigenvalues of $\mathbf{A}$ have negative real parts. An internally stable system is also externally stable. Lyapunov and Poincaré proved that a nonlinear system is stable with respect to small perturbations from an equilibrium $\mathbf{x}_0$ if the system matrix of the system linearized about $\mathbf{x}_0$ is stable.

2.6 Control

In the broadest sense, control is the augmentation of a given dynamical system by additional systems in order to achieve a desired overall behavior. The most common control

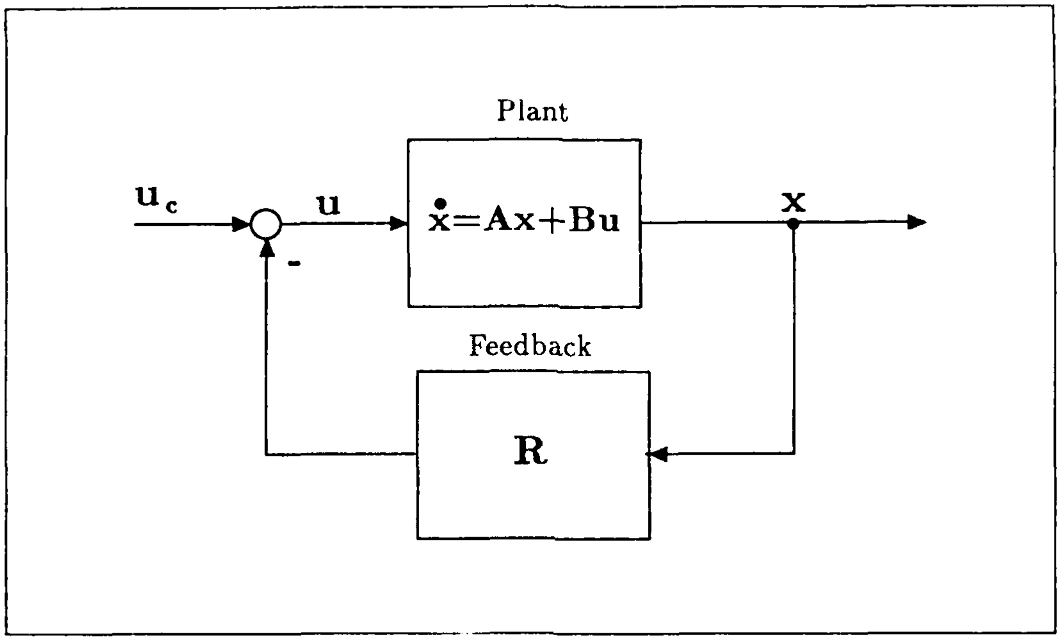


Figure 3: Feedback control

structure is feedback where the state $\mathbf{x}$ is transformed by a compensator matrix $\mathbf{R}$, compared with a commanded input vector $\mathbf{u}_c$ and the difference used as input $\mathbf{u}$ to the plant as in figure (3).

We then have that

$$\mathbf{u} = \mathbf{u}_c - \mathbf{R}\mathbf{x} \tag{17}$$

which gives us the overall plant equation of

$$\dot{\mathbf{x}} = (\mathbf{A} - \mathbf{B}\mathbf{R})\mathbf{x} + \mathbf{R}\mathbf{u}_c \tag{18}$$

To influence the overall behavior of this system we may choose the matrix $\mathbf{R}$. The properties of a dynamic system are determined by the position of the eigenvalues of the system matrix (cf. stability). A common criterion for the choice of the elements of $\mathbf{R}$ is therefore to achieve desired locations of the poles of $\mathbf{A} - \mathbf{B}\mathbf{R}$. This is called *pole placement.*

2.7 Observers and Filters

We are often interested in knowing the state $\mathbf{x}(t)$ of our dynamical system but can only measure the output $\mathbf{y}(t)$, for instance in the case of feedback control. An observer is a dynamical system that reconstructs the state $\mathbf{x}(t)$ given the output $\mathbf{y}(t)$ and the input $\mathbf{u}(t)$ when the dynamics of plant and output are known (figure (4)).

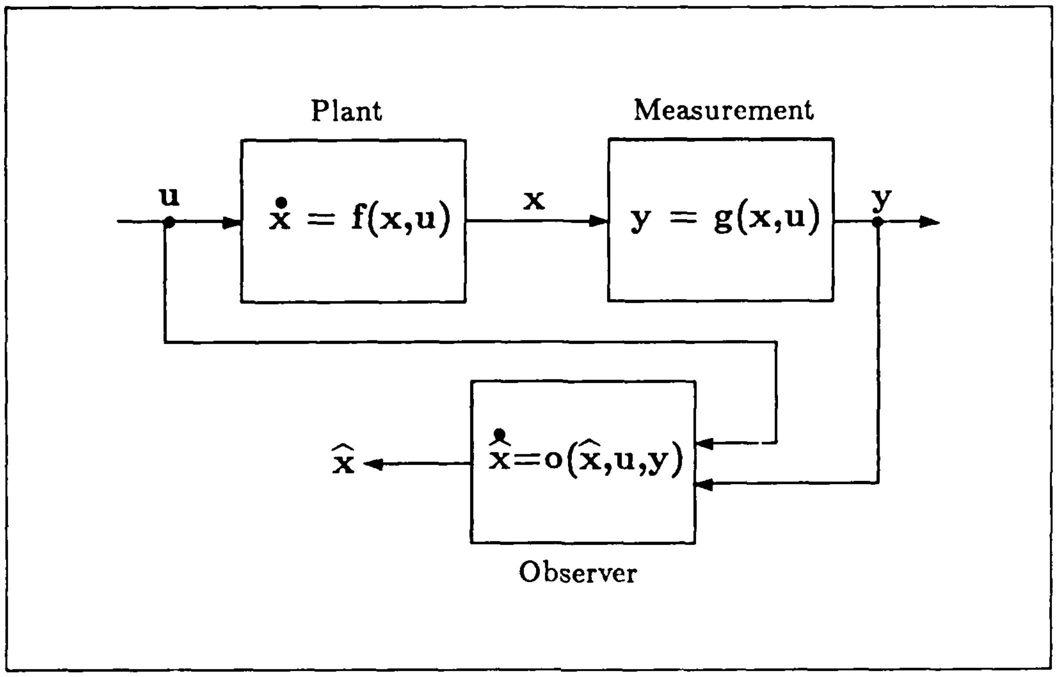


Figure 4: *The observer principle*

In the linear case (3), (4) we use the following observer structure

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{E}(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}} - \mathbf{D}\mathbf{u}) \tag{19}$$

where $\hat{\mathbf{x}}$ denotes our estimate for the state vector $\mathbf{x}$ and $\mathbf{E}$ is a matrix that we may choose freely. We see that this is a copy of the original plant equation augmented by a term for the error between estimated and actual measurement value. A criterion for the choice of $\mathbf{E}$ may be derived from the error $\varepsilon = \mathbf{x} - \hat{\mathbf{x}}$. From (19) and substituting (4) for $\mathbf{y}$ we have

$$\dot{\varepsilon} = (\mathbf{A} - \mathbf{E}\mathbf{C})\varepsilon \tag{20}$$

which means that the dynamics of the error depend on the eigenvalues of $\mathbf{A} - \mathbf{E}\mathbf{C}$. We may therefore influence the dynamics of this error using pole placement techniques on this matrix. We will at least require this matrix to be asymptotically stable in order for the error to disappear eventually. For discrete systems the error can vanish after a finite number of sampling periods (dead beat response). This can be achieved by placing all poles of the discrete observer at the origin.

A *Kalman filter* is derived from an observer of a linear system as in (3), (4) where state and output are corrupted by Gaussian noise of a known distribution. State, input and output are interpreted as random processes with a known distribution of the initial state vector $\mathbf{x}(0)$. The filter then consists of two stages for every iteration: a prediction phase in which the next state vector is computed by using the current state estimate in the state equations and an update phase in which the current estimate is altered according to the error between expected and actual measurement just as we have seen for observers. The key difference between observers and Kalman filters is that the gain matrix **E** depends on the variance of the measurements, or in other words, a discrepancy between predicted and actual measurement influences the estimate in proportion to the confidence we have in that measurement. This provides the filter with the ability to estimate optimally the state vector in the sense of minimal noise corruption, but places the burden of determining the variances on the designer. This methodology may be extended to nonlinear systems by linearizing the state and output equations around the current estimate in every iteration. This is known as the Extended Kalman Filter (refer to [16]). We have summarized the equations of the Extended Kalman Filter below. Observe that they reduce to a simpler form in the special case of linear system.

The Extended Kalman Filter (21)

System Model	$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k$	$\mathbf{w}_k \sim N(\mathbf{0}, \mathbf{Q}_k)$
Output Model	$\mathbf{y}_k = \mathbf{g}(\mathbf{x}_k) + \mathbf{v}_k$	$\mathbf{v}_k \sim N(\mathbf{0}, \mathbf{R}_k)$
Initial State	$\mathbf{x}_0 \sim N(\hat{\mathbf{x}}_0, \mathbf{P}_0)$	
Non-correlation	$E[\mathbf{w}_k \mathbf{v}_k^T] = \mathbf{0}$ for all k	
State Estimate Propagation	$\hat{\mathbf{x}}_{k+1}^- = \mathbf{f}(\hat{\mathbf{x}}_k^+, \mathbf{u}_k)$	
Error Covariance Propagation	$\mathbf{P}_{k+1}^- = \mathbf{\Phi}_k \mathbf{P}_k^+ \mathbf{\Phi}_k^T + \mathbf{Q}_k$	
State Estimate Update	$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{E}_k[\mathbf{y}_k - \mathbf{g}(\hat{\mathbf{x}}_k^-)]$	
Error Covariance Update	$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{E}_k \mathbf{C}_k^T)\mathbf{P}_k^-$	
Filter Gain	$\mathbf{E}_k = \mathbf{P}_k^- \mathbf{C}_k [\mathbf{C}_k^T \mathbf{P}_k^- \mathbf{C}_k + R_k]^{-1}$	

We have used the abbreviations $\mathbf{\Phi}_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k}$ for the system Jacobian, $\mathbf{C}_k = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k}$ for the output Jacobian and $x \sim N(\mu, \sigma)$ for a normally distributed random variable x with expected value μ and variance σ. The Kalman filter forms the core of the estimation algorithms for motion vision presented below.

3 Dynamical Systems for Motion Vision

Before formally modelling the imaging situation as a dynamical system we provide some insight into why such an interpretation may apply. Recall that a dynamical system consists of a state that changes in time according to a plant equation (1) and a measurable output which is a function of that state at every instant (2). In the imaging situation in motion vision the position and attitude of an object being viewed changes in time according to the kinematic equations. Only a function of position and attitude, namely the projection into the image plane, can be measured at every instant. We could identify position and attitude as the states of our system, the kinematic equations as the plant and the 2D projection as our measurement.

However, the interpretation is not quite that simple. We are not really measuring the position and orientation of the 2D projection. We merely have an array of brightness values available in every frame. One approach would then be to determine first the location and orientation of the projection from the grey-level array and use the result as a new "meta"-measurement. This leads to the feature-based approaches mentioned in the introduction. An alternative is to make assumptions about the formation of the brightness value and in particular how a change in brightness relates to a change in position and orientation. The most common assumption is the brightness change constraint equation of Horn and Schunck [27]. The idea there is that the brightness of a patch corresponding to a fixed location on the surface being viewed will not change significantly from one image to the next.

A conclusion we can draw from these thoughts is that the major emphasis in finding suitable dynamical models for the motion imaging situation will be on constructing appropriate measurement models. Since there is no unique model for a given system a major task is to investigate different models and identify those that represent the temporal behavior most accurately while remaining at a low level of complexity (nonlinearities, order of the system, etc.).

We have constructed three models of the imaging situation that differ mainly in the type of measurement assumed. All models include the depth (distance to the object) as a state that is not measurable in the output. The main idea is to employ a state space observer or a Kalman filter as described in section 2.7 to estimate the depth. One would also like to recover the motion of object or observer. Note that an observer/filter solution would then require the presence of some plant equation describing the temporal evolution of the motion parameters. In the case of object motion this is rarely ever known and we can only hypothesize a model as done in section 5. When the camera is in motion we may use the dynamics of the actuating device to obtain such a relationship as we suggest in section 7. We believe that the best procedure is to estimate depth with a Kalman filter and use the resulting depth map to compute motion in a least-squares fashion. This is the approach taken in section 6.

The first model we present is in a sense the physically "correct" model as it attempts to establish the relationship between the measured brightness value variation and the object/observer motion. This requires an assumption about the reflectance properties of the surface (lambertian, specular etc.) and is therefore rather restrictive. We also assume that motion is known. A dense depth map is recovered using an Extended Kalman Filter.

The second model employs the brightness change constraint assumption to avoid modeling the surface reflectance and illumination conditions explicitly. Since the brightness change constraint equation is an implicit equation, this requires a modification of the general observer scheme. The other important characteristic of this approach is that the surface being viewed is assumed to be planar, which allows us to parametrize it and recover the surface parameters rather than a dense depth map. We also show how the use of a simple motion model permits us to recover motion within the observer.

Section 6 finally presents a system that estimates both motion and depth recursively. A dense depth map is obtained either directly from brightness values or from a correlation-based displacement measurement using the current estimate for the motion parameters.

This constitutes the update section of a non-linear Kalman filter. The new depth map is then used to find a new least-squares estimate for the current motion and a predicted depth map for the subsequent frame. This is formulated as the prediction stage of a Kalman filter. This scheme requires no assumptions about surface reflectance, shape, illumination, or motion (besides the ones inherent to the brightness constancy assumption, see Verri and Poggio [60]) and is therefore best suited for implementation and processing of real imagery.

The remainder of this section is devoted to the introduction of basic terminology for the imaging situation, choice of coordinate system, etc., which is similar to that of Horn [25] and Negahdaripour [42].

3.1 Coordinate Systems

We will use a coordinate system with origin coincident with the center of projection of the camera, the z-axis pointing towards the image plane. The image plane is parallel to the x-y-plane at unit distance from the origin i.e. we express image-plane coordinates in units of focal length. The situation is shown in figure (5).

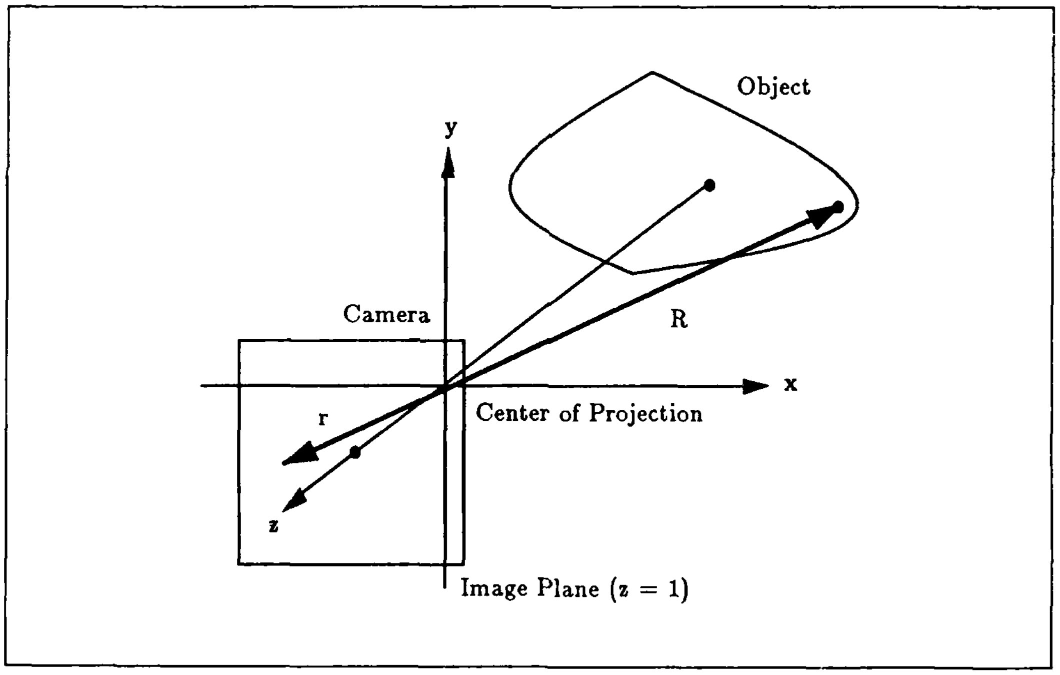


Figure 5: The viewer centered coordinate system

A point on an object in the real world is represented by the vector

$$\mathbf{R} = [X, Y, Z]^T \tag{22}$$

and the projection of that point into the image plane by

$$\mathbf{r} = [x, y, 1]^T. \tag{23}$$

The perspective projection equation contains the relationship between both points:

$$\mathbf{r} = \frac{\mathbf{R}}{\mathbf{R} \cdot \hat{\mathbf{z}}} = \frac{\mathbf{R}}{Z} \tag{24}$$

where $\hat{\mathbf{z}}$ denotes a unit vector along the z-axis.

3.2 Continuous Motion

We assume that object and camera are in relative motion given by a translational vector $\mathbf{t}(t)$ and a rotational vector $\omega(t)$ in the coordinate system of the camera. In [18] we showed that the motion perceived in the image plane is the relative motion of object and camera if both object and camera motion parameters are defined in a coordinate system of the camera. We also showed that the generalization from instantaneous velocities $\mathbf{t}$, ω to velocity functions of time $\mathbf{t}(t)$, $\omega(t)$ is valid.

Consider a point on the observed object with position vector $\mathbf{R}(t)$. The motion of the point on the object is given by

$$\dot{\mathbf{R}}(t) = -\mathbf{t}(t) - \omega(t) \times \mathbf{R}(t) \tag{25}$$

and the corresponding motion field in the image plane is

$$\dot{\mathbf{r}} = -\hat{\mathbf{z}} \times (\mathbf{r} \times (\mathbf{r} \times \omega - \frac{\mathbf{t}}{\mathbf{R} \cdot \hat{\mathbf{z}}})). \tag{26}$$

This is also referred to as the *motion field* equation.

3.3 Discrete Motion

We will discover that although the above continuous formulation accounts precisely for the changes due to motion, a discrete description is better suited for the purpose of constructing a computer algorithm to implement dynamical systems techniques. This means that we must use finite rotations and translations instead of the infinitesimal quantities $\mathbf{t}$ and ω. There are numerous ways of representing finite interframe transformations. We will use a displacement vector $\mathbf{p}$ for the translation and a unit quaternion $\mathring{q}$ for the rotation. Unit quaternions are an efficient means of representing rotations, used in vision by [7], [22], [24]. Information on the subject is available in [51], [53], [22] and [48] where the latter provides an excellent comparison of various representations of rotation.

For our purposes it will be sufficient to understand a unit quaternion, which we will denote by a circle above a character in what follows, as a four-dimensional unit vector or a generalized complex number

$$\mathring{q} = \begin{bmatrix} q_0 \\ q_x \\ q_y \\ q_z \end{bmatrix} = q_0 + iq_x + jq_y + kq_z. \tag{27}$$

Rotations about an axis along the unit vector $\hat{\omega} = [\omega_x, \omega_y, \omega_z]^T$ by an angle θ are represented by the unit quaternion

$$\overset{\circ}{q} = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}(i\omega_x + j\omega_y + k\omega_z). \tag{28}$$

Vectors are purely imaginary, scalars purely real quaternions. So if $\overset{\circ}{q}$ denotes a rotation as above and $\overset{\circ}{x}$ is the quaternion corresponding to some vector $\mathbf{x}$ then the rotated quaternion is

$$\overset{\circ}{x}' = \overset{\circ}{q}\overset{\circ}{x}\overset{\circ}{q}^* = \begin{bmatrix} \overset{\circ}{q}\cdot\overset{\circ}{q} & 0 & 0 & 0 \\ 0 & (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_xq_y - q_0q_z) & 2(q_xq_z + q_0q_y) \\ 0 & 2(q_xq_y - q_0q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_yq_z - q_0q_x) \\ 0 & 2(q_xq_z - q_0q_y) & 2(q_yq_z - q_0q_x) & (q_0^2 - q_x^2 - q_y^2 + q_z^2) \end{bmatrix} \tag{29}$$

where $\overset{\circ}{q}^*$ is the conjugate quaternion to $\overset{\circ}{q}$.

How does a rotation and translation in this representation change a real-world position vector $\mathbf{R}_k$ amd its quaternion equivalent $\overset{\circ}{R}_k$ at discrete time k? We rotate $\overset{\circ}{R}_k$ by a quaternion $\overset{\circ}{q}_k$ using (29) and translate by a finite displacement quaternion $\overset{\circ}{p}_k$ to obtain $\overset{\circ}{R}_{k+1}$:

$$\overset{\circ}{R}_{k+1} = \overset{\circ}{p}_k + \overset{\circ}{q}_k\overset{\circ}{R}_k\overset{\circ}{q}_k^* \tag{30}$$

The discrete motion field (the displacement field $\overset{\circ}{r}_{k+1} - \overset{\circ}{r}_k$) is determined by

$$\overset{\circ}{r}_{k+1} = \frac{\overset{\circ}{R}_{k+1}}{Z_{k+1}} = \frac{\overset{\circ}{p}_k + (\overset{\circ}{q}_k\overset{\circ}{r}_k\overset{\circ}{q}_k^*)Z_k}{((\overset{\circ}{p}_k + (\overset{\circ}{q}_k\overset{\circ}{r}_k\overset{\circ}{q}_k^*)Z_k)\cdot\overset{\circ}{z})} \tag{31}$$

where $\overset{\circ}{z}$ is a unit quaternion with z-component 1. The denominator of the last term is simply the last component of equation (30).

4 A Discrete Model for Brightness Measurements

The first decision when modeling a dynamical system is which quantities should be included in the state vector and which quantities are measurable as output. We have already indicated the importance of the measurement model, which we derive first in this section. We designate the brightness values $E(x, y)$ to be the measurements in this model where x and y are image plane coordinates. The measurement model must then relate $E(x, y)$ to the time-varying quantities in the svstem such as the location and orientation of the object being viewed. Depending on which relationship we obtain, we must then compile all time-varying quantities therein into the state vector and formulate a plant equation that governs their temporal evolution.

Another major decision is whether to formulate the model discretely or continuously. Since the measurements are taken discretely and the Kalman filter is inherently a discrete algorithm, we decided to formulate the model discretely.

4.1 An Output Equation for Brightness Values

We have designated the brightness values in the image to be our output or measurement. To derive a corresponding output equation for use in our dynamical system we must determine the relationship between the brightness E at some point (x, y) in the image plane and the time-varying parameters of the system such as position and attitude, which we later will include in the state equations. We need some basic physical facts about the imaging situation that we have taken from [25], chapter 10.

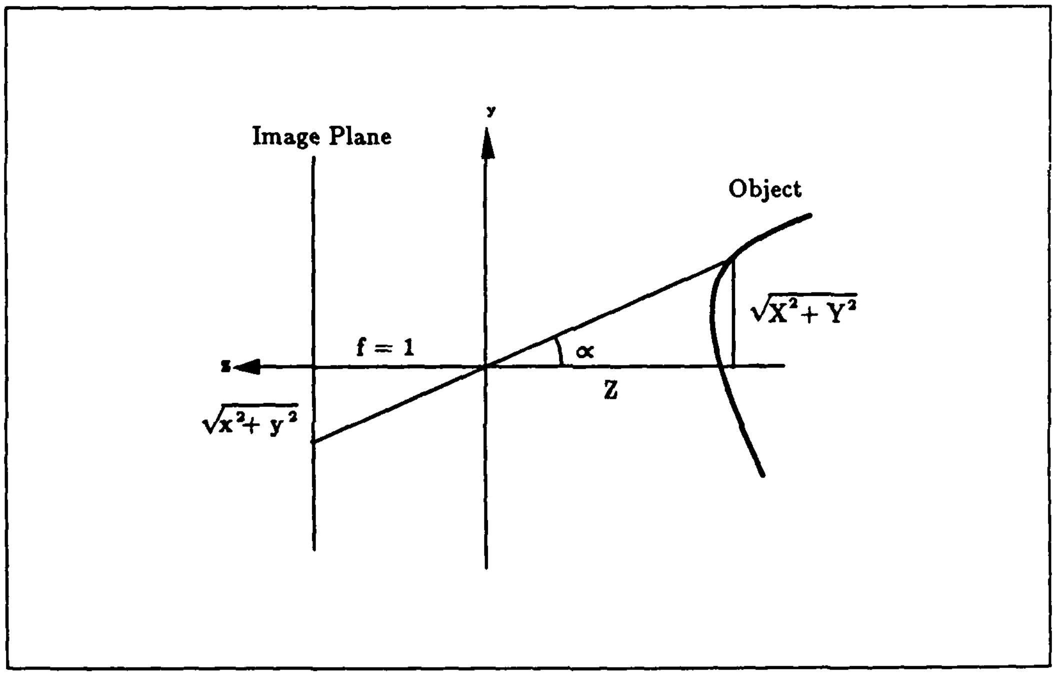


Figure 6: The formation of brightness values

Consider a point P on an object whose position vector forms an angle α with the optical axis and whose radiance is L (figure 6). The image brightness at the projection of P into the image plane is

$$E = L\frac{\pi}{4}(\frac{d}{f})^2 \cos^4 \alpha \tag{32}$$

where d is the diameter of the camera lens and f is the focal length which we scale to 1 in what follows. The ccsine can be expressed in terms of the real-world or image coordinates

$$\cos \alpha = \frac{Z}{\sqrt{X^2 + Y^2 + Z^2}} = \frac{1}{\sqrt{1 + x^2 + y^2}} \tag{33}$$

so we have

$$E = \frac{L\pi d^2 Z^4}{4(X^2 + Y^2 + Z^2)^2}. \tag{34}$$

Unfortunately, L depends on the type of surface being viewed and the illumination. On the other hand it is intuitive that we will not be able to model image brightness values correctly without some assumptions about illumination and surface reflectance. The two most widely used surface models are *Lambertian* and *specular*. For the Lambertian surface we have (see for instance [25])

$$L = \rho(\mathbf{i} \cdot \mathbf{n}) \tag{35}$$

where ρ is the surface albedo, $\mathbf{i}$ is a unit vector in the direction of the light source and $\mathbf{n}$ is a unit vector along the surface normal of the point being viewed.

Several models have been proposed for specular surfaces. The one developed by Tuong-Phong [57] and applied to computer vision by Horn [26], [23] models the radiance of a surface illuminated by an extended source as

$$L = \lambda(\frac{\mathbf{R} \cdot \mathbf{s}}{|\mathbf{R}|})^n \tag{36}$$

where λ is a specular reflectance coefficient, n a parameter that describes the compactness of the specular patch, and $\mathbf{s}$ is a unit vector in the direction of perfect specular reflectance which can be written as

$$\mathbf{s} = 2(\mathbf{i} \cdot \mathbf{n})\mathbf{n} - \mathbf{i}. \tag{37}$$

In practice a linear combination

$$L = tL_{lambertian} + (1 - t)L_{specular} \tag{38}$$

provides a good approximation in which t may vary spatially.

Using the expressions for the radiance (35), (36) in the equation for the image brightness (34) yields

$$E = \frac{\rho\pi d^2 Z^4}{4|\mathbf{R}|^4}(\mathbf{i} \cdot \mathbf{n}) \tag{39}$$

and

$$E = \frac{\lambda\pi d^2 Z^4}{4|\mathbf{R}|^{4+n}}(2(\mathbf{i} \cdot \mathbf{n})(\mathbf{R} \cdot \mathbf{n}) - (\mathbf{i} \cdot \mathbf{R}))^n. \tag{40}$$

We see that in both cases E depends on the time-varying quantities $\mathbf{R}$ (the position vector) and $\mathbf{n}$ (the surface normal). This means we must include these variables in our state vector.

We note that in both the specular and the Lambertian case, we can use $\mathbf{r}$ instead of $\mathbf{R}$ in the expressions for E. The time dependency of $\mathbf{r}$ is more complex than the one for $\mathbf{R}$ as we see by comparing (25) and (26). For this reason and for the sake of variety we use $\mathbf{R}$ here. We also focus on the simpler Lambertian case in what follows, although the specular case is analogous.

4.2 The Model

Our output equation (39) dictates that the state vector must contain at least $\mathbf{R}$ and $\mathbf{n}$. We must therefore determine how these vectors change due to motion. Our experience with the previous model revealed the advantages of a discrete motion representation. We assume that the camera undergoes a translation $\overset{\circ}{p}_k$ and rotation $\overset{\circ}{q}_k$ at time k. What is the state equation for $\overset{\circ}{R}_k$, i.e. how does $\overset{\circ}{R}_{k+1}$ depend on $\overset{\circ}{R}_k$? This is simply the object motion equation (30):

$$\overset{\circ}{R}_{k+1} = \overset{\circ}{p}_k + \overset{\circ}{q}_k \overset{\circ}{R}_k \overset{\circ}{q}{}^*_k. \tag{41}$$

We must determine the analogous relationship for the change in the unit surface normal $\overset{\circ}{n}_k$. A unit normal remains unchanged under translation of the coordinate system - it will merely be rotated by $\overset{\circ}{q}$ according to

$$\overset{\circ}{n}_{k+1} = \overset{\circ}{q}_k \overset{\circ}{n}_k \overset{\circ}{q}{}^*_k. \tag{42}$$

We can now summarize our model from (41), (42) and the output from (39) as

$$\mathbf{x}_{k+1} = \begin{bmatrix} \overset{\circ}{R}_{k+1} \\ \overset{\circ}{n}_{k+1} \end{bmatrix} = \begin{bmatrix} \overset{\circ}{p}_k + \overset{\circ}{q}_k \overset{\circ}{R}_k \overset{\circ}{q}{}^*_k \\ \overset{\circ}{q}_k \overset{\circ}{n}_k \overset{\circ}{q}{}^*_k \end{bmatrix} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \tag{43}$$

$$y_k = E_k = \frac{\rho \pi d^2 Z_k^4}{4|\mathbf{R}_k|^4}(\mathbf{i} \cdot \mathbf{n}_k) = g(\mathbf{x}_k) \tag{44}$$

where the input is $\mathbf{u}_k = [\overset{\circ}{p}_k, \overset{\circ}{q}_k]^T$. This model contains some redundancy since $\overset{\circ}{R}$, $\overset{\circ}{p}$ and $\overset{\circ}{n}$ are vectors, i.e. purely imaginary quaternions (we can omit their first components in the model), $\mathbf{n}$ is a unit vector (we only need two of its components) and $\overset{\circ}{q}$ is a unit quaternion (we only need three components to represent it). So a minimal model could have the state and input

$$\mathbf{x}_k = \begin{bmatrix} X_k \\ Y_k \\ Z_k \\ n_{xk} \\ n_{yk} \end{bmatrix} \qquad \mathbf{u}_k = \begin{bmatrix} p_{xk} \\ p_{yk} \\ p_{zk} \\ q_{xk} \\ q_{yk} \\ q_{zk} \end{bmatrix}. \tag{45}$$

This dynamical system describes the change in brightness for the projection of a particular point at position vector $\mathbf{R}$ as a result of translation $\overset{\circ}{p}$ and rotation $\overset{\circ}{q}$. As before, we are interested in recovering the depth Z_k and finding a filter suitable for performing this task.

4.3 Estimating Depth using an Extended Kalman Filter

The extended Kalman filter is an observer for a nonlinear system corrupted by noise with certain probabilistic assumptions about the distribution of the noise and the initial values. The filter equations are summarized in table (21).

We assume that our system (43) is corrupted by noise $\mathbf{w}_k$ which is $N(\mathbf{0}, \mathbf{Q}_k)$ distributed, so that

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k. \tag{46}$$

Similarly, the measurements are influenced by noise v_k which is $N(0, r_k)$ distributed. The measurement equation becomes

$$y_k = g(\mathbf{x}_k) + v_k. \tag{47}$$

We finally assume that the initial value of the state $\mathbf{x}_0$ is normally distributed around our initial guess $\hat{\mathbf{x}}_0$ with known covariance $\mathbf{P}_0$ and that the noise processes influencing state and output are uncorrelated: $E[\mathbf{w}_k v_k] = \mathbf{0}$ for all k.

We can make these assumptions in our particular case but must deal with some of the issues explicitly before proceeding. A good initial state estimate i.e. $\hat{\mathbf{x}}_0$, is not available without additional information. We can only guess it, but we can run our filter with several different initial values and select those that converge. Similarly, the covariance $\mathbf{P}_0$ of the initial state estimate is unknown and must be guessed. The covariance $\mathbf{Q}$ of the state, however, may be assumed to be zero since the kinematic equations are geometric relationships and not subject to noise. We may have some knowledge about the uncertainty in our velocity information which enters into the kinematic equations. This in turn would result in a non-zero covariance. The measurement variance $\mathbf{r}$ is due to noise in the camera sensor. A noise model and the corresponding variance is presented in section 6.

We find that the state estimate propagates as

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{f}(\hat{\mathbf{x}}_k^+, \mathbf{u}_k) \tag{48}$$

and the covariance as

$$\mathbf{P}_{k+1}^- = \mathbf{\Phi}_k \mathbf{P}_k^+ \mathbf{\Phi}_k^T + \mathbf{Q}_k. \tag{49}$$

Between propagation we update the estimates from $\hat{\mathbf{x}}_k^-$ to $\hat{\mathbf{x}}_k^+$ according to

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{e}_k[y_k - g(\hat{\mathbf{x}}_k^-)] \tag{50}$$

and the covariance according to

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{e}_k \mathbf{c}_k^T)\mathbf{P}_k^- \tag{51}$$

where the Kalman filter gain is computed as

$$\mathbf{e}_k = \mathbf{P}_k^- \mathbf{c}_k [\mathbf{c}_k^T \mathbf{P}_k^- \mathbf{c}_k + r_k]^{-1}. \tag{52}$$

These equations contain the Jacobians $\mathbf{\Phi}_k = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_k}$ and $\mathbf{c}_k = \frac{\partial g}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}_k}$.

In order to apply this algorithm we need merely compute the matrices $\mathbf{\Phi}$ and $\mathbf{c}$. In the case of the minimal system with state and input from (45) we have

$$\mathbf{\Phi} = \begin{bmatrix} (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_x q_y - q_0 q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_0 q_x) \\ 2(q_x q_z - q_0 q_y) & 2(q_y q_z - q_0 q_x) & (q_0^2 - q_x^2 - q_y^2 + q_z^2) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{53}$$

$$\begin{array}{cc} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ (q_0^2+q_x^2-q_y^2-q_z^2) & 2(q_xq_y-q_0q_z) \\ 2(q_xq_y-q_0q_z) & (q_0^2-q_x^2+q_y^2-q_z^2) \end{array}\Bigg] \tag{54}$$

and

$$\mathbf{c} = \frac{\rho\pi d^2 Z^4}{|\mathbf{R}|^4}\begin{bmatrix} \frac{\mathbf{i}\cdot\mathbf{n}}{|\mathbf{R}|^2}X \\ \frac{\mathbf{i}\cdot\mathbf{n}}{|\mathbf{R}|^2}Y \\ \frac{\mathbf{i}\cdot\mathbf{n}}{|\mathbf{R}|^2}\frac{X^2+Y^2}{Z} \\ \frac{i_x}{4} \\ \frac{i_y}{4} \end{bmatrix}. \tag{55}$$

where we have omitted the hat as an indication of estimation and the time index k.

Let us finally consider how the measurement and filter process will be implemented. The plant equation (43) describes how a point P and the surface normal at that point change relative to the camera due to motion. The Kalman filter algorithm will therefore recover the depth Z of that particular point. Consequently, the brightness measurement used in the filter must be the brightness at the location of the projection of P into the image plane. Since this projected point moves throughout the measurement process, we would have to try to estimate the position of the point in every frame and then take the measurement there, which seems rather error-prone. To obtain a dense depth map, we must repeat this for all points in the region of interest.

Instead, we will store the current depth estimate with every grid point of the image array and use the brightness value at that location to update the state estimate in the Kalman filter. The new depth estimate will then be associated with the location $(\hat{x},\hat{y}) = (\hat{X}/\hat{Z},\hat{Y}/\hat{Z})$ which is the position we expect the projection of P to move to according to our estimate. This position may not lie at a grid point so we must interpolate to obtain the values at the grid points. The details of this interpolation are discussed in section 6. We can summarize the filter algorithm (there is one filter for every point observed) as follows:

(1) **Set initial values:**
For every point 1 to m under consideration set $\hat{\mathbf{x}}_0^+ = 0$ unless better initial guess available.

(2) **Filter loop:**
For frames $k = 1$ to n do

For all points 1 to m do

(2.1) Measure the brightness values E_k.

(2.2) Compute the system matrices Φ_k and $\mathbf{c}_k$ from equations (54) and (55) for the current state estimate $\hat{\mathbf{x}}_k^+$ and the known input $\mathbf{u}_k$.

(2.3) Compute the state estimate $\hat{\mathbf{x}}_{k+1}^-$ and the covariance $\mathbf{P}_{k+1}^-$ for time $k+1$ from equations (48), (49).

(2.4) Compute the filter gain e_{k+1} from equation (52).

(2.5) Update the state estimate to $\hat{x}^{+}_{k+1}$ and the covariance to P^{+}_{k+1} using equations (50), (51).

(2.6) Associate the new state with image location $(\hat{X}^{+}_{k+1}/\hat{Z}^{+}_{k+1}, \hat{Y}^{+}_{k+1}/\hat{Z}^{+}_{k+1})$ and interpolate the state estimates at integral grid point locations.

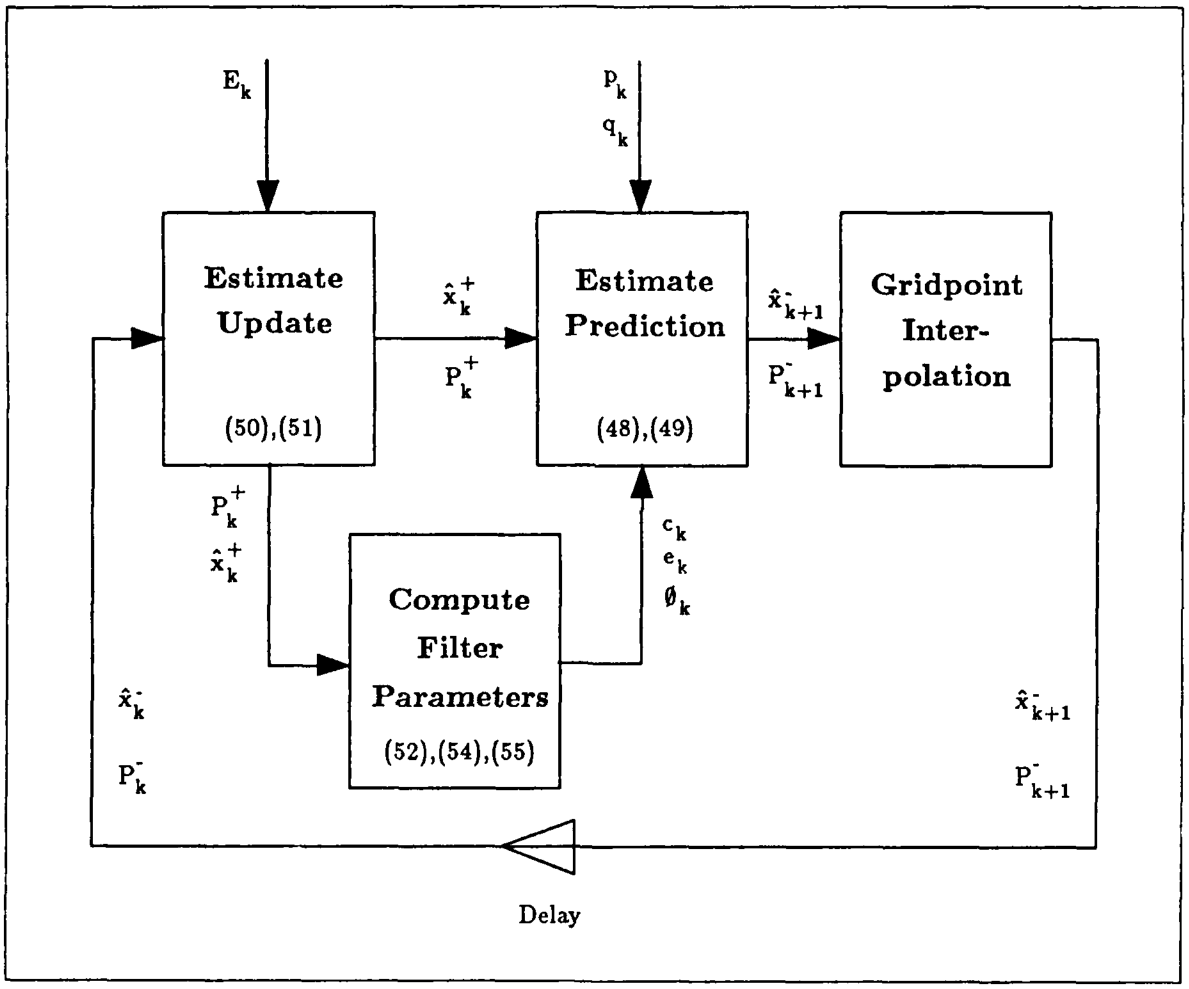


Figure 7: A block diagram of the Kalman filter

A block-diagram of this algorithm is shown in figure (7). There is one such filter loop for every point in the region of interest.

4.4 Discussion

We can easily see that the above algorithm runs in $\Theta(nm)$ time where n is the number of frames and m is the number of points. This is clearly optimal for surface estimation that makes no assumption about the surface structure (i.e. we must look at every point in every frame). A parallel implementation of the algorithm could assign one processor to every image plane pixel, which can compute the observer matrices for those points in parallel. Every processor will also evaluate the new state estimate, which determines the location of the processor to operate on the pixel in the next frame. The current estimate must be transmitted to that processor. Since interframe changes will be small for high frame rates it is sufficient that locally neighboring processors be interconnected in order to execute this transmission efficiently. The parallel complexity is then merely $\Theta(n)$.

So besides being fast, the algorithm makes no assumptions about surface structure and can produce a dense depth map. The *interesting properties (as well as the deficiencies)* of the algorithm given above are summarized here:

(1) We use a discrete model and avoid discretization errors.

(2) The use of brightness values as output permits us to operate directly on the measured data but also requires knowledge of surface and illumination parameters.

(3) We assume that motion is known.

(4) Noise is modelled.

(5) Depth is recovered using a nonlinear Kalman filter that is optimal in the presence of noise.

(6) Efficiency and perhaps convergence rely on a good initial guess for the state vector.

From this perspective we can also see why the filter algorithms given by Broida and Chellappa [6], [7] require feature correspondences between frames to be precomputed. Indeed if we had those correspondences here, the performance of the algorithm would increase significantly because we no longer have to estimate the position to which a given point will move in the next frame and interpolate the state estimates at grid points. Conversely the estimator can be used to help establish feature correspondences because we estimate where the feature has moved to in the subsequent frame. This is an additional interesting application. Only tests with an implementation will eventually reveal performance and accuracy. Problems (1) and (2) are inherent to the model and we attempt to alleviate them in what follows.

5 A Model for a Planar Surface using the Brightness Change Constraint Equation

We have seen that the attempt to construct a precise physical model of the imaging situation forces us to make assumptions that cannot always be guaranteed for real images (for instance that we have lambertian surfaces). We will use the approximation of the brightness change constraint equation in this section to avoid having to model the formation of brightness values in the image explicitly via surface reflectance properties. Another improvement the previous model calls for is to drop the requirement of known motion. This is addressed by assuming a very simple dynamical model of the motion which allows us to incorporate motion into the plant equation. Finally, we assume that a planar surface is being viewed which changes the nature of the problem from having to estimate a dense depth map to estimating a small number of parameters for the plane.

5.1 The Discrete BCCE: An Implicit Output Equation

Horn and Schunck popularized the use of the brightness change constraint equation (BCCE)

$$\frac{dE}{dt} = 0 \tag{56}$$

in their work on optical flow [27]. Negahdaripour, Weldon and Horn [29], [42] employed the BCCE to recover motion and surface parameters instantaneously. Here we investigate the use of the BCCE as an output equation of a dynamical system. The BCCE can be expressed in terms of image brightness derivatives

$$\mathbf{E}_r \cdot \dot{\mathbf{r}} + E_t = 0 \tag{57}$$

where $\mathbf{E}_r = [\frac{\partial E}{\partial x}, \frac{\partial E}{\partial y}, 0]^T$ and $E_t = \frac{\partial E}{\partial t}$. We wish to obtain a discrete form of the BCCE analogous to the previous equation. The Taylor series expansion of E provides

$$\Delta E = \mathbf{E}_r \cdot \Delta\mathbf{r} + E_t \Delta t = 0 \tag{58}$$

and assuming that we have measurements at discrete times $k = 0, 1, \ldots$ we obtain $\Delta\mathbf{r} = \mathbf{r}_{k+1} - \mathbf{r}_k$ and $\Delta t = k + 1 - k = 1$ so

$$\mathbf{E}_{rk} \cdot (\mathbf{r}_{k+1} - \mathbf{r}_k) + E_{tk} = 0. \tag{59}$$

The discrete motion field $\mathring{r}_{k+1}$ is given in equation (31) in section 3.3. We can use this equation by reexpressing the above BCCE in terms of unit quaternions. This presents no difficulty since all vectors are simply replaced by corresponding purely imaginary quaternions. Substituting (31) into (59) yields

$$\mathring{E}_{rk} \cdot \left(\frac{\mathring{p}_k + (\mathring{q}_k \mathring{r}_k \mathring{q}_k^*) Z_k}{((\mathring{p}_k + (\mathring{q}_k \mathring{r}_k \mathring{q}_k^*) Z_k) \cdot \mathring{z})} - \mathring{r}_k \right) + E_{tk} = 0 \tag{60}$$

and after some rearrangement we have

$$(\mathring{q}_k \mathring{r}_k \mathring{q}_k^* + \frac{\mathring{p}_k}{Z_k}) \cdot \mathring{s}_k = 0 \tag{61}$$

in which we have abbreviated $\mathring{s}_k = \mathring{E}_{rk} + (E_{tk} - \mathring{E}_{rk} \cdot \mathring{r}_k)\mathring{z} = [0, E_{xk}, E_{yk}, E_{tk} - x_k E_{xk} - y_k E_{yk}]^T$. This is the *discrete brightness change constraint equation.*

In this model we restrict ourselves to planar surfaces. The equation of such a plane is

$$\mathbf{r} \cdot \mathbf{n} = \frac{1}{\mathbf{R} \cdot \hat{\mathbf{z}}} = \frac{1}{Z} \tag{62}$$

where $\mathbf{n}$ is a normal to the surface and $1/|\mathbf{n}|$ is the distance of the plane to the origin. Substituting the quaternion equivalent into the discrete BCCE yields

$$(\mathring{q}_k \mathring{r}_k \mathring{q}_k^* + (\mathring{n}_k \cdot \mathring{r}_k)\mathring{p}_k) \cdot \mathring{s}_k = 0. \tag{63}$$

This relationship depends on a particular point in the image plane. Since we wish to use the BCCE to make statements about the plane and its motion that are obviously independent of particular points in the image plane, the question arises at which point in the image plane (63) should be evaluated. Since no point is distinguished we may choose one at random. A better approach is to sum up the values over a region of the plane. Of course then we must square them before adding to avoid cancellation. This could be written as

$$\sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}_k \mathring{r}_k \mathring{q}_k^* + (\mathring{n}_k \cdot \mathring{r}_k)\mathring{p}_k) \cdot \mathring{s}_k]^2 = 0 \tag{64}$$

where w and h are height and width of the image region under consideration.

We see that this equation implicitly relates the measurable quantities $\mathring{s}_k$ to the motion and surface parameters $\mathring{p}$, $\mathring{q}$ and $\mathring{n}$ at every image plane point $\mathbf{r}$. So if we interpret the former as an output

$$\mathbf{y} = \mathbf{s} \tag{65}$$

and the latter as a state

$$\mathbf{x} = \begin{bmatrix} \mathbf{n} \\ \mathbf{p} \\ \mathbf{q} \end{bmatrix} \tag{66}$$

we have a relationship of the form

$$g(\mathbf{x}, \mathbf{y}) = 0. \tag{67}$$

We have used vector symbols here to denote the imaginary parts of the above quaternions. Since $\mathring{s}$, $\mathring{p}$ and $\mathring{n}$ are purely imaginary and $\mathring{q}$ is a unit quaternion it suffices to include the three imaginary components in the state - the remaining component is either zero or related by some algebraic equation. Unlike our usual output equation (2), equation (67) only implicitly relates state and output so that standard observation techniques do not apply. Before we consider modifications of our observer to deal with this problem we have yet to establish the state equations.

5.2 The State Equations for the Moving Plane

How do the three components $\mathring{n}$, $\mathring{p}$ and $\mathring{q}$ of our state vector change in time? Let us first determine how the surface normal will change due to the motion. This is not trivial since we use the inverse length of $\mathring{n}$ to express the distance from the origin.

For pure rotation, the distance between plane and origin does not change and the normal is simply rotated by $\overset{\circ}{q}_k$:

$$\overset{\circ}{n}_{k+1} = \overset{\circ}{q}_k \overset{\circ}{n}_k \overset{\circ}{q}{}^*_k. \tag{68}$$

For pure translation, the orientation of the normal remains unchanged but the distance changes by the amount of the translation displacement $\overset{\circ}{p}_k$ projected onto the unit normal along $\overset{\circ}{n}$ so

$$\frac{1}{|\overset{\circ}{n}_{k+1}|} = \frac{1}{|\overset{\circ}{n}_k|} + \overset{\circ}{p}_k \cdot \frac{\overset{\circ}{n}_k}{|\overset{\circ}{n}_k|} \tag{69}$$

which can be rearranged to

$$|\overset{\circ}{n}_{k+1}| = \frac{|\overset{\circ}{n}_k|}{1 + \overset{\circ}{p}_k \cdot \overset{\circ}{n}_k}. \tag{70}$$

Since a unit normal along $\overset{\circ}{n}$ would remain unchanged by translation, we have

$$\overset{\circ}{n}_{k+1} = \frac{\overset{\circ}{n}_k}{|\overset{\circ}{n}_k|}|\overset{\circ}{n}_{k+1}| = \frac{\overset{\circ}{n}_k}{1 + \overset{\circ}{p}_k \cdot \overset{\circ}{n}_k}. \tag{71}$$

In this model we will not interpret the motion parameters $\overset{\circ}{p}$ and $\overset{\circ}{q}$ as input but as states. This allows us to recover them with our observer but we must make some assumption about the dynamics of these vectors in order to formulate a state equation. In a general case we could say that $\overset{\circ}{p}_k$ and $\overset{\circ}{q}_k$ are the solution to some simple difference equation

$$\overset{\circ}{p}_{k+1} = \mathbf{P}\overset{\circ}{p}_k \qquad \text{and} \qquad \overset{\circ}{q}_{k+1} = \mathbf{Q}\overset{\circ}{q}_k \tag{72}$$

which leaves the problem of finding the elements of **P** and **Q**. In the case of a moving camera these matrices are given by the dynamics of the actuating device (mobile robot, motion platform etc.). We believe that this interpretation is worth pursuing since it has many practical applications. Here we will content ourselves with an approximation. We assume that the motion vectors will not change significantly between frames because the sampling rate is high compared to the kinematic changes so that the matrices **P** and **Q** are null matrices. We then have a special form of (72):

$$\overset{\circ}{p}_{k+1} = \overset{\circ}{p}_k \qquad \text{and} \qquad \overset{\circ}{q}_{k+1} = \overset{\circ}{q}_k. \tag{73}$$

We can now summarize our state equations from (71), (73) and the implicit output equation from (64)

$$\mathbf{x}_{k+1} = \begin{bmatrix} \overset{\circ}{n}_{k+1} \\ \overset{\circ}{p}_{k+1} \\ \overset{\circ}{q}_{k+1} \end{bmatrix} = \begin{bmatrix} \frac{\overset{\circ}{n}_k}{1+\overset{\circ}{p}_k \cdot \overset{\circ}{n}_k} \\ \overset{\circ}{0} \\ \overset{\circ}{0} \end{bmatrix} = \mathbf{f}(\mathbf{x}_k) \tag{74}$$

$$g(\mathbf{x}_k, \mathbf{y}_k) = g(\begin{bmatrix} \mathring{n}_k \\ \mathring{p}_k \\ \mathring{q}_k \end{bmatrix}, \mathring{s}_k) = \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}_k \mathring{r}_k \mathring{q}_k^* + (\mathring{n}_k \cdot \mathring{r}_k)\mathring{p}_k) \cdot \mathring{s}_k]^2 = 0. \quad (75)$$

As we already mentioned, we can restrict ourselves to the imaginary components of the quaternions in the above relationships.

5.3 A Nonlinear Observer for the Moving Plane

It is clear that if we succeed in constructing an observer for this system we can recover depth and motion. However, the output equation is not of the desired form and depends on the image plane point at which the brightness gradients are measured. The solution to this problem is to modify the observer such that it can handle the implicit output equation.

Let us recapitulate the basic idea behind the observer in section 2.7. The observer was basically a copy of the dynamical system under observation corrected by a matrix multiple of the error between actual and estimated output. In our case a discrepancy between the "real" state $\mathbf{x}$ and the estimated state $\hat{\mathbf{x}}$ will result in a non-zero value for $g(\hat{\mathbf{x}}, \mathbf{y})$. So we can simply use $g(\hat{\mathbf{x}}, \mathbf{y})$ as an error term in the discrete version of (19):

$$\hat{\mathbf{x}}_{k+1} = \mathbf{f}(\hat{\mathbf{x}}_k) + \mathbf{e}g(\hat{\mathbf{x}}_k, \mathbf{y}_k). \quad (76)$$

The estimation error $\varepsilon_k = \mathbf{x}_k - \hat{\mathbf{x}}_k$ is found to be

$$\varepsilon_{k+1} = \mathbf{f}(\mathbf{x}_k) - \mathbf{f}(\hat{\mathbf{x}}_k) + \mathbf{e}(g(\mathbf{x}_k, \mathbf{y}_k) - g(\hat{\mathbf{x}}_k, \mathbf{y}_k)) \quad (77)$$

where we have exploited $g(\mathbf{x}_k, \mathbf{y}_k) = 0$ to maintain symmetry. The nonlinearity of state and output however prevent us from applying the simple rules presented in section 2.7 for the calculation of $\mathbf{e}$. Using the ideas of the extended Kalman fil· we expand the non-linearities about the current estimate

$$\mathbf{f}(\mathbf{x}_k) = \mathbf{f}(\hat{\mathbf{x}}_k) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k} (\mathbf{x}_k - \hat{\mathbf{x}}_k) + \ldots \approx \mathbf{f}(\hat{\mathbf{x}}_k) + \mathbf{A}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k) \quad (78)$$

$$g(\mathbf{x}_k, \mathbf{y}_k) = g(\hat{\mathbf{x}}_k, \mathbf{y}_k)) + \left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k, \mathbf{y}} (\mathbf{x}_k - \hat{\mathbf{x}}_k) + \ldots \approx g(\hat{\mathbf{x}}_k, \mathbf{y}_k)) + \mathbf{c}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k) \quad (79)$$

which we substitute into equation (77):

$$\varepsilon_{k+1} = \mathbf{A}_k \varepsilon_k - \mathbf{e}(\mathbf{c}_k \cdot \varepsilon_k) = (\mathbf{A}_k - \mathbf{e}\mathbf{c}_k^T)\varepsilon_k. \quad (80)$$

The error difference equation is time-varying and in order to achieve a good dynamic behavior we must adjust $\mathbf{e}$ with every measurement, so we have $\mathbf{e} = \mathbf{e}_k$. The idea is then, to compute $\mathbf{A}_k$ and $\mathbf{c}_k$ for every new estimate $\hat{\mathbf{x}}_k$, then select $\mathbf{e}_k$ such that the eigenvalues of $\mathbf{A}_k - \mathbf{e}_k\mathbf{c}_k^T$ are located at the origin and finally compute the new estimate $\hat{\mathbf{x}}_{k+1}$. The matrices $\mathbf{A}$ and $\mathbf{c}$ have been computed in the appendix A. We then have the following algorithm:

(1) **Set initial values:**
Set $\hat{\mathbf{x}}_0 = \mathbf{0}$ (unless better information is available).

(2) **Observer loop:**
For frames $k = 1$ to n do

(2.1) **Compute system matrices:**
Compute $\mathbf{A}_k$ and $\mathbf{c}_k$ using the formulas (139) and (141) for the current estimate $\hat{\mathbf{x}}_k$ and measurement $\mathbf{y}_k$.

(2.2) **Compute observer gain:**
Compute the vector $\mathbf{e}_k$ such that all eigenvalues of $\mathbf{A}_k - \mathbf{e}_k\mathbf{c}_k^T$ are zero.

(2.3) **Compute next estimate:**
Compute the next estimate $\hat{\mathbf{x}}_{k+1}$ from equation (76).

Since we are not using a filter, the algorithm becomes simpler which is reflected in the block-diagram, figure (8).

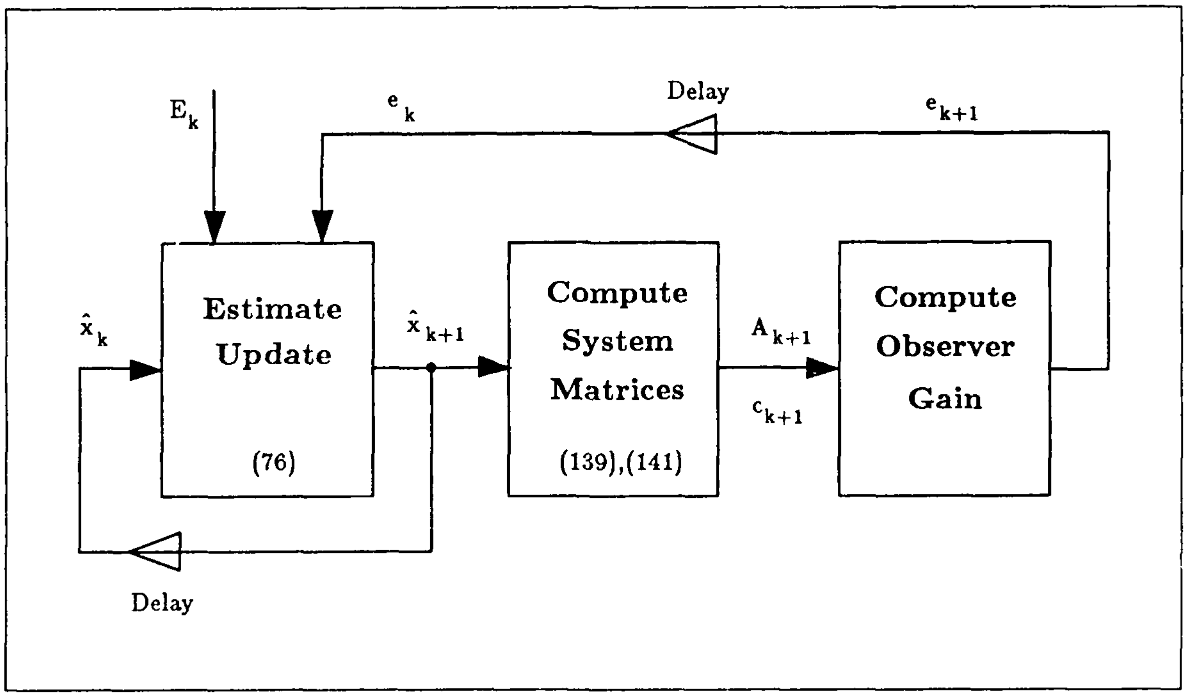


Figure 8: A block diagram of the observer algorithm

5.4 Discussion

As before, we would like to make a statement about the complexity of this observer algorithm. Here, however, we do not have m observers, one for each point of interest. But we have an image region of width $2w$ and height $2h$ over which we are sampling. If again we denote the number of frames by n, the algorithm will run in $\Theta(whn)$ since one measurement for every image point must be processed to compute the vector **c** in (2.1) in every frame. Comparing the complexities is rather useless since the results and assumptions are quite different from those in the preceding algorithms. A parallel implementation that provides an efficient summation of the expressions in the components of **c** could reach a parallel complexity of $\Theta(n)$.

We also note that a closed form solution for the instantaneous problem (2 frames) using velocities **t** and ω exists (cf. Negahdaripour and Horn [42]). It may be used to provide a good initial value for the observation algorithm. On the other hand, an observer algorithm as given above must compete with the repeated application of the instantaneous algorithm. Applying Negahdaripour's solution to n frames also requires $\Theta(whn)$ operations. A comparison of accuracy can take place once implementations of both schemes are available. In doing this, we must observe one important detail: The closed-form solution has been shown only for instantaneous velocities, not finite displacements and rotations as given in this section. If an iterative scheme is necessary to recover the latter from two frames, this would clearly increase the complexity.

We summarize important properties of the above algorithm:

(1) Incremental recovery of surface orientation and motion parameters.

(2) Applicable only to planar surfaces.

(3) Motion parameters are assumed to be constant over time.

(4) The model uses finite translations and rotations represented by quaternions.

(5) The discrete brightness change constraint equation is formulated and employed as an implicit output equation.

(6) A nonlinear observer is presented to handle implicit output equations.

From the theoretical point of view, this is the most appealing of the models presented in this paper as it recovers surface *parameters* rather than distinct depth values. On the other hand it only applies to the special case of a plane in motion. Perhaps a fusion of both will provide interesting results.

6 Integrated Motion and Depth Estimation

The previous examples have shown the duality between motion and depth: If we know the depth, we can recover the motion and vice versa. But if we try to solve both problems simultaneously we must impose constraints that restrict the solutions to rather specialized cases. A conclusion might be to separate the two computations and feed one with the result of the other in an iterative scheme.

We have seen further that the Kalman filter is not particularly well suited for motion estimation since no dynamical model of the behavior of the motion parameters is available. The idea in this section is therefore to estimate depth using the Kalman filter and update the motion estimate to fit the depth map in a least-squares sense in every iteration.

The algorithms in this section make no assumptions about surface or motion other than the ones inherent to the BCCE and are therefore most likely to be successful on real imagery. For this reason we discuss all the necessary details in-depth as for instance measurement variances and pixel interpolation. We have tailored the scheme to two different implementation environments: one in which only brightness values are available as measurements and the other in which a correlation-based displacement estimator is used. Conceptually both approaches follow the same idea but they have different efficiency and stability properties.

6.1 Using brightness measurements directly

The brightness change constraint equation is the foundation of this approach. We use it in its differential form

$$\frac{dE}{dt} = \mathbf{E}_r \cdot \dot{\mathbf{r}} + E_t = 0 \tag{81}$$

where $\dot{\mathbf{r}}$ is the motion field from equation (26). If we substitute the motion field for $\dot{\mathbf{r}}$ we obtain

$$c + \mathbf{v} \cdot \omega + \frac{1}{Z}(\mathbf{s} \cdot \mathbf{t}) = 0 \tag{82}$$

after some rearrangement (refer to Negahdaripour and Horn [42] for the derivation). We have used the following abbreviations: $c = E_t$, $\mathbf{s} = (\mathbf{E}_r \times \hat{\mathbf{z}}) \times \mathbf{r}$ and $\mathbf{v} = -\mathbf{s} \times \mathbf{r}$. The reader will have noticed that we have again resorted to the continuous case. The reason for this lies in the fact that the brightness change constraint equation is simpler in this form and yields a closed-form solution for the motion when depth is known.

We must also specify the dynamical system that will be used. It will be one-dimensional with the sole state-variable $Z(t)$. The plant equation is simply the third component of the kinematic equation (25) i.e.

$$\dot{Z} = t_z + (\omega_x y - \omega_y x)Z. \tag{83}$$

Z will also be our measurement so we have a very simple model. There is a slight problem with this formulation, however, which lies in the fact that x and y also vary in time and should be included in the state. We avoid this by interpreting them as time-variant parameters that must be updated in every iteration according to the motion-field equation (26). What this means is that the depth prediction Z according to the above plant equation will be valid at the location specified by the new coordinates (x', y') which may not lie on a grid point of the image array. This leads to the problem of having to interpolate the depth as we mentioned in section 4. We discuss the treatment of these problems in detail below.

Now consider the block diagram of figure (9). A new image arrives at iteration k and is fed into the depth estimator. Using the motion estimate from $k-1$ we can solve the brightness change constraint equation (82) for Z at every image point and obtain a dense

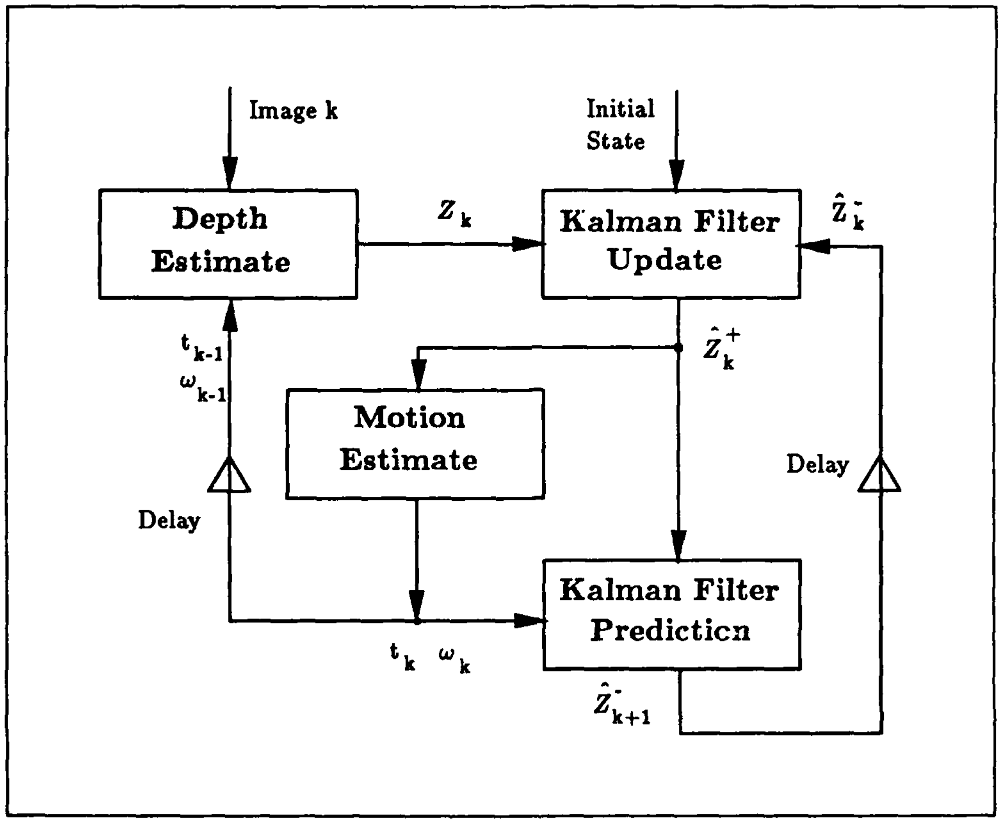


Figure 9: Integrated depth and motion estimation based on brightness values

depth measurement. This is used in the update stage of a Kalman filter to produce a depth estimate. Note that in this case the state variable being estimated and the measurement quantity are identical. The Kalman filter is merely used as an update algorithm.

The updated depth map is used to compute the motion. We compute the motion such that the error in the brightness change constraint equation (the deviation from zero) is minimized in a least-squares sense over the region of interest. Finally we use the Kalman filter prediction to compute the expected depth map in the next measurement which is made available to the filter's update stage. We now discuss the various modules in detail.

6.1.1 Depth Estimation

Estimating depth proceeds in a straightforward manner based on the brightness change constraint equation (82). We solve for Z and obtain

$$Z = -\frac{\mathbf{s} \cdot \mathbf{t}}{c + \mathbf{v} \cdot \omega}. \tag{84}$$

The quantities c, **v** and **s** depend on the spatial brightness gradients E_x and E_y and the temporal brightness gradient E_t which can be approximated using finite differences of the image brightness values. The motion parameters **t** and ω are taken from the previous motion estimate. We thereby obtain a dense depth map.

Since this Z will be the measurement quantity in our Kalman filter, we also need its variance. A closer look at (84) reveals that noise in the measurement of the brightness values will affect Z. Let us recall a basic result of probability theory. Suppose we have a random variable Z which is a function of a collection of random variables $x_1, \ldots, x_n$ so that

$$Z = f(x_1, \ldots, x_n). \tag{85}$$

Using a Taylor series expansion of f around the "true" value of Z we find that

$$\sigma_Z^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\partial f}{\partial x_i} \frac{\partial f}{\partial x_j} cov(x_i, x_j) \tag{86}$$

where $cov(x_i, x_j)$ denotes the covariance of the two random variables. Now if we assume that the x_i are independent, which is usually the case in measurement processes, we have $cov(x_i, x_j) = 0$ for $i \neq j$ and $cov(x_i, x_j) = \sigma_{x_i}^2$ for $i = j$. We then have

$$\sigma_Z^2 = \sum_{i=1}^{n} (\frac{\partial f}{\partial x_i})^2 \sigma_{x_i}^2. \tag{87}$$

In our case the brightness measurements are interpreted as random variables and we are interested in determining how the variance in the brightness measurements will affect the variance of the depth estimate. We must find the immediate dependency of depth and brightness in equation (82). We substitute the expressions for c, **v** and **s** into (82) and find

$$Z = \frac{E_x(t_x - xt_z) + E_y(t_y - yt_z)}{E_t + E_x(xy\omega_x - (1 + x^2)\omega_y + y\omega_z) + E_y((1 + y^2)\omega_x - xy\omega_y - x\omega_z)}. \tag{88}$$

We abbreviate the coefficients of the brightness gradients to obtain a more compact expression

$$Z = \frac{aE_x + bE_y}{E_t + cE_x + dE_y}. \tag{89}$$

The brightness gradients are estimated through some finite difference approximation as for instance the one suggested by Horn and Schunck [27] :

$$\begin{aligned}
E_x &\approx \tfrac{1}{4d_x}(E(x + d_x, y, t) - E(x, y, t) + E(x + d_x, y + d_y, t) - E(x, y + d_y, t) + \\
&\quad E(x + d_x, y, t + T) - E(x, y, t + T) + E(x + d_x, y + d_y, t + T) - \\
&\quad E(x, y + d_y, t + T)) \\
E_y &\approx \tfrac{1}{4d_y}(E(x, y + d_y, t) - E(x, y, t) + E(x + d_x, y + d_y, t) - E(x + d_x, y, t) + \\
&\quad E(x, y + d_y, t + T) - E(x, y, t + T) + E(x + d_x, y + d_y, t + T) - \\
&\quad E(x + d_x, y, t + T)) \\
E_t &\approx \tfrac{1}{4T}(E(x, y, t + T) - E(x, y, t) + E(x, y + d_y, t + T) - E(x, y + d_y, t) + \\
&\quad E(x + d_x, y, t + T) - E(x + d_x, y, t) + E(x + d_x, y + d_y, t + T) - \\
&\quad E(x + d_x, y + d_y, t)).
\end{aligned} \tag{90}$$

We have denoted the distance between two pixels in the x direction by d_x, in the y direction by d_y and the inverse frame rate by T. Note that this approximation is the average over 4 first differences along the edges of a cube in spatio-temporal hyperspace. The approximation is valid at the center of that cube.

We can apply our formula (87) for the propagation of variances to calculate the variance in Z. Note, however, that the E_x, E_y and E_t are not independent and can therefore not be used. We must express Z in terms of the E_i (the brightness values used in the gradient approximations (90)) which we may assume to be independent identically distributed random variables with variance σ_E^2. We then find the variance in Z to be

$$\sigma_Z^2 = \sum_{i=1}^{8}(\frac{\partial Z}{\partial E_i})^2\sigma_E^2 = (\frac{\partial Z}{\partial E_x}\sum_{i=1}^{8}\frac{\partial E_x}{\partial E_i} + \frac{\partial Z}{\partial E_y}\sum_{i=1}^{8}\frac{\partial E_y}{\partial E_i} + \frac{\partial Z}{\partial E_t}\sum_{i=1}^{8}\frac{\partial E_t}{\partial E_i})^2\sigma_E^2 \tag{91}$$

which evaluates to

$$\sigma_Z^2 = \frac{1}{2}\sigma_E^2\frac{(aE_t + (ad - bc)E_y)^2\frac{1}{d_x^2} + (bE_t + (bc - ad)E_x)^2\frac{1}{d_y^2} + (aE_x + bE_y)^2\frac{1}{T^2}}{(E_t + cE_x + dE_y)^4}. \tag{92}$$

We have determined how the variance in the depth is related to the variance in the measurements. But what is the variance in the measurements? One component of the measurement noise can be modeled: the quantization noise. Since our sensor discretizes brightness values into gray-levels, values that do not coincide with a discretization step will be "rounded off" to the next step. In our model of the sensor, the discrete sensor output values will be denoted by E_i and the constant quantization step by $\Delta E = E_{i+1} - E_i$. If we assume that our sensor discretizes in the following fashion

$$E_{sensor} = \begin{cases} E_{i+1} & \text{if} \quad E_{actual} > E_i + \Delta E/2 \\ E_i & \text{if} \quad E_{actual} \leq E_i + \Delta E/2 \end{cases} \tag{93}$$

and that brightness values are equally likely in $[E_i, E_{i+1}]$ the expected sensor output is easily determined as $(E_{i+1} + E_i)/2$ and the variance as

$$\sigma_E^2 = \frac{(\Delta E)^2}{4}. \tag{94}$$

However, this models only one possible source of noiseand distortion. Others include the electrical circuitry in the sensor or simply blurring or defocusing. So instead of attempting to model all these various noise sources, we can simply conduct a measurement in which a uniformly colored simple surface is viewed to provide a set of measurements $E_1, \ldots, E_n$. We then determine the expected value

$$E(E) = \frac{1}{n}\sum_{i=1}^{n} E_i \tag{95}$$

and the variance

$$\sigma_E^2 = \frac{1}{n}\sum_{i=1}^{n}(E_i - E(E))^2. \tag{96}$$

Using these results we can obtain a dense depth map where every depth value is accompanied by a variance that will be used by the filter we describe below.

6.1.2 Kalman Filter Update of Depth Map

In order to update our estimate $\hat{Z}_k$ we must first compute the filter gain from (21). Since the system is one-dimensional, this is a scalar

$$e_k = p_k^-(p_k^- + r_k)^{-1} = \frac{p_k^-}{p_k^- + r_k} \tag{97}$$

where r_k is the variance of the measurement. We are measuring the depth so $r_k = \sigma_Z^2$ which was computed in the previous subsection.

The filter update equations from (21) are also extremely simple:

$$\begin{aligned} \hat{Z}_k^+ &= \hat{Z}_k^- + e_k(Z_k - \hat{Z}_k^-) & (98) \\ p_k^+ &= (1 - e_k)p_k^-. & (99) \end{aligned}$$

We now have a new depth map and also a covariance map. The former is used in the motion estimator described below.

6.1.3 Motion Estimation

This module must solve the following problem: given the depth at every image point, what is the global relative motion $\mathbf{t}$, ω between camera and environment. Under the assumption of the validity of the brightness change constraint equation, this problem has been solved [29]. Assuming that the region of interest has width $2w$ and height $2h$ centered at the origin, we are seeking a motion estimate $\mathbf{t}$ *and* ω *that minimizes*

$$\sum_{x=-w}^{w} \sum_{y=-h}^{h} (c + \mathbf{v} \cdot \omega + \frac{1}{Z}(\mathbf{s} \cdot \mathbf{t}))^2. \tag{100}$$

Differentiating with respect to $\mathbf{t}$ and ω and equating to $\mathbf{0}$ results in

$$\begin{aligned} (\sum_{x=-w}^{w} \sum_{y=-h}^{h} (\mathbf{v}\mathbf{s}^T)\frac{1}{Z})\omega \quad &+ \quad (\sum_{x=-w}^{w} \sum_{y=-h}^{h} (\mathbf{s}\mathbf{s}^T)\frac{1}{Z^2})\mathbf{t} \quad = \quad -\sum_{x=-w}^{w} \sum_{y=-h}^{h} \frac{c}{Z}\mathbf{s} \\ (\sum_{x=-w}^{w} \sum_{y=-h}^{h} (\mathbf{v}\mathbf{v}^T))\omega \quad &+ \quad (\sum_{x=-w}^{w} \sum_{y=-h}^{h} (\mathbf{s}\mathbf{v}^T)\frac{1}{Z})\mathbf{t} \quad = \quad -\sum_{x=-w}^{w} \sum_{y=-h}^{h} c\mathbf{v} \end{aligned} \tag{101}$$

a 6x6 system that can be solved for the desired parameters in general.

6.1.4 Kalman Filter Prediction of Depth Map

Given the current estimate for the motion parameters and the updated estimate for the depth map the prediction phase of the Kalman filter produces a predicted depth map for the following iteration. Looking at the Kalman filter (21) we see that we need predictions for state and covariance. A simple first differences approximation for our plant (83) yields the following discrete form

$$Z_{k+1} = Z_k + T(t_z + (\omega_x y - \omega_y x)Z_k) \tag{102}$$

which is the prediction equation for the depth. We can see that the system "matrix" is

$$\phi_k = 1 + T(\omega_x y - \omega_y x). \tag{103}$$

It is needed in the prediction equation for the covariance which reads

$$p_{k+1}^- = [1 + T(\omega_x y - \omega_y x)]^2 p_k^+. \tag{104}$$

As we have argued previously we have assumed that the plant equation of the system is not noise corrupted as it is purely kinematic. Therefore $q_k = 0$.

This procedure, however, is not quite correct. We have neglected the fact that the new estimate $\hat{Z}_{k+1}^-$ will not be valid at the same image plane location (x, y) at which $\hat{Z}_k^+$ was stored since x and y change over time. To make things worse, the position at which our estimate will be valid may not coincide with a grid point of the image array. So we must determine where the point at which we are estimating the depth moves to and interpolate the depth at the grid points from this transformed depth map.

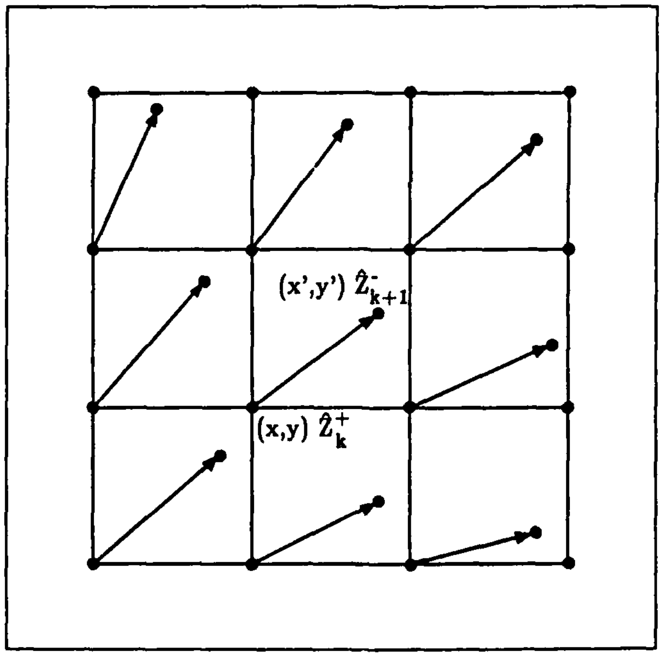


Figure 10: Displacement of observed points between successive images

The movement of a point (x, y) in the image plane is described by the motion-field equation (26). A finite differences discretization yields

$$x' = x + T(\frac{-t_x + x t_z}{Z} + \omega_x x y - \omega_y (x^2 + 1) + \omega_z y) \tag{105}$$

$$y' = y + T(\frac{-t_y + y t_z}{Z} - \omega_y x y + \omega_x (y^2 + 1) - \omega_z x) \tag{106}$$

which provides us with the coordinates (x', y') at which our predicted depth Z^-_{k+1} is valid. We can think of this as a new depth map which evolves from the old one by the prediction process as depicted in figure 10. The remaining problem is to interpolate the depth and the variance at the grid points of the image array. We investigate two methods for accomplishing this in which we focus on the depth interpolation.

The first method simply fits a plane to the data in a least-squares fashion. Supposing the depth values Z_i at coordinates (x_i, y_i) for $i = 1, \ldots, n$ are within some neighborhood of the grid-point we are considering. We locally approximate the real-world surface by a plane

$$aX + bY + cZ = 1 \tag{107}$$

and using the perspective projection equations we have

$$axZ + byZ + cZ = 1. \tag{108}$$

We seek to determine the parameters a, b and c such that the deviation of our depth values from the resulting plane

$$\sum_{i=1}^{n} (ax_iZ_i + by_iZ_i + cZ_i - 1)^2 \tag{109}$$

is minimized. The minimum can be found by differentiating and equating to zero. This yields the following 3x3 system

$$\begin{array}{ccccccc} a\sum_{i=1}^{n}(x_iZ_i)^2 & + & b\sum_{i=1}^{n} x_iy_iZ_i^2 & + & c\sum_{i=1}^{n} x_iZ_i^2 & = & \sum_{i=1}^{n} x_iZ_i \\ a\sum_{i=1}^{n} x_iy_iZ_i^2 & + & b\sum_{i=1}^{n}(y_iZ_i)^2 & + & c\sum_{i=1}^{n} y_iZ_i^2 & = & \sum_{i=1}^{n} y_iZ_i \\ a\sum_{i=1}^{n} x_iZ_i^2 & + & b\sum_{i=1}^{n} y_iZ_i^2 & + & c\sum_{i=1}^{n} Z_i^2 & = & \sum_{i=1}^{n} Z_i \end{array} \tag{110}$$

which can be solved for a, b and c. Then the depth value at the grid-point x_0, y_0 is easily computed as

$$Z_0 = -\frac{1}{ax_0 + by_0 + c}. \tag{111}$$

The method is rather intensive in terms of computational cost. We must compute 9 different sums over all of the n measurement values and solve a 3x3 linear system for ever interpolation point. We only obtain an interpolation for a planar approximation.

An alternative that is more computationally oriented is to compute the grid-point value as some weighted average of the estimates in its neighborhood

$$Z(x', y') = \sum_{i=1}^{n} w_iZ(x_i, y_i). \tag{112}$$

We have set up some criteria for the choice of the weighting function w_i which can be found in appendix B. We show there that

$$w_i = \frac{\frac{1}{d_i^2}}{\sum_{i=1}^{n} \frac{1}{d_i^2}}. \tag{113}$$

fulfills a set of natural requirements that one would demand of such a weighted average. The appendix also contains a linear-time algorithm for computing the weights and the average so that the resulting scheme is very efficient.

Other interpolation schemes include bi-linear and bi-cubic interpolation. See Rifman and McKinnon [46], Bernstein [4] or Abdou and Wong [1] for some interesting and practically valuable techniques.

6.1.5 Discussion

Our scheme for integrated motion and depth estimation is now complete. The characteristics and deficiencies of this scheme are summarized below.

(1) We recover a dense depth map using a Kalman filter and a motion estimate to fit the depth map in a least-squares sense.

(2) The validity of the brightness change constraint equation is assumed. No other assumptions about surface, reflectance or motion are made.

(3) We need no optical flow or displacement estimator - the algorithm operates directly on brightness values.

(4) Estimates are based on gradient approximations that tend to have little numerical robustness.

(5) Depth and motion estimates rely on the same physical relationship. This may lead to instability of the iterative estimation.

Item (4) is an inherent property of all schemes that rely on the differential form of the brightness change constraint equation. An alternative is to consider an equivalent integral constraint to obtain higher robustness. The 6th item reflects the "chicken-and-egg" problem that arises when we sequence motion and depth estimation rather than recover them simultaneously. Since the latter cannot be achieved in general, we rely on the filter to provide sufficient convergence to render the effect of this problem negligible. We address these concerns by slightly altering the implementation as described below.

6.2 Using SSD-based displacement estimates as measurements

This section investigates an alternative measurement procedure to the one based on the differential form of the brightness change constraint equation described in the previous section. In order to remedy the problems with the differential approach (refer to items

5 and 6 in the discussion) we require two main properties of an alternative formulation: measurements should be based on information from a larger image area (an integral rather than a differential approach) and the measurement process should be independent of the motion estimation.

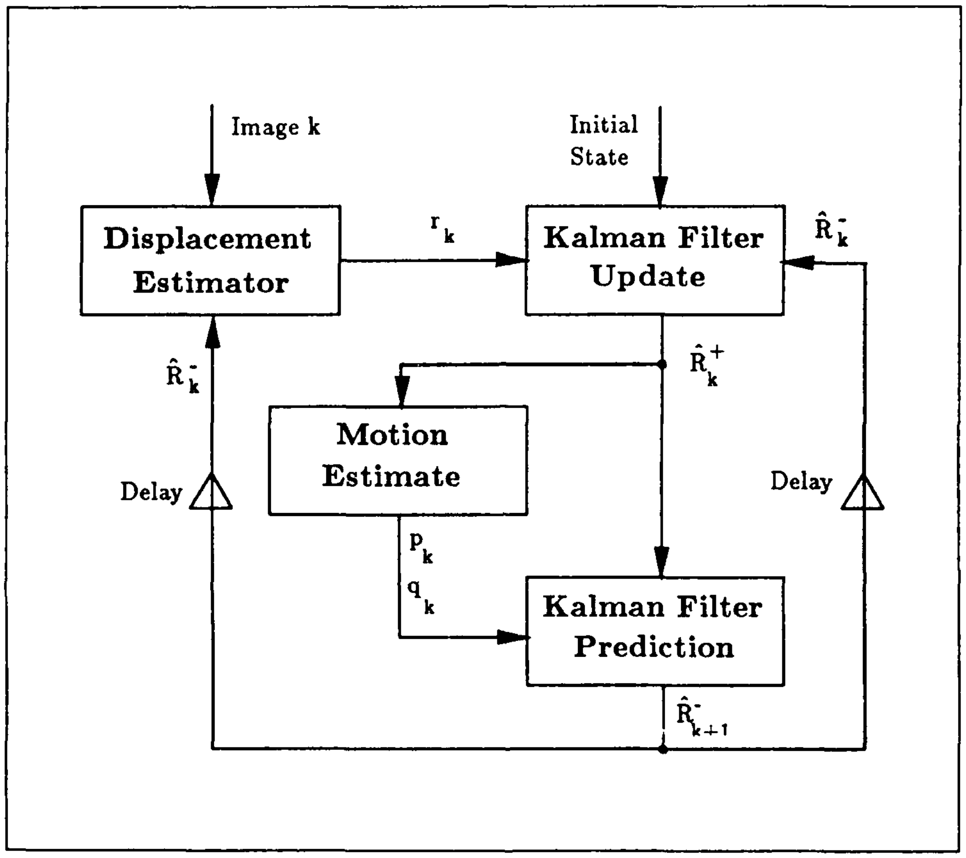


Figure 11: Integrated depth and motion estimation based on correlation-based displacement measurements

Figure (11) has basically the same structure as the one in the previous subsection. We have replaced the depth estimator by an sum-of-squared-differences (SSD) displacement estimator. The output of this system is a vector $\mathbf{r}_k = (x_k, y_k)$ for every image point which describes where a given pixel in the previous image has moved to in the current image. Given these measurements we are forced to change the structure of our dynamic system to produce these quantities as output. We resort to our initial formulation of the object kinematics and have chosen the discrete formulation (30) for the sake of variety. We have

$$\overset{\circ}{R}_{k+1} = \overset{\circ}{p}_k + \overset{\circ}{q}_k \overset{\circ}{R}_k \overset{\circ}{q}_k^* \tag{114}$$

where $\overset{\circ}{R}$ denotes the real world position of a point on the surface being viewed and $\overset{\circ}{p}, \overset{\circ}{q}$ represent translation and rotation. The components of our plant are therefore

$$X_{k+1} = p_x + q_{22}X_k + q_{23}Y_k + q_{24}Z_k \quad (115)$$
$$Y_{k+1} = p_y + q_{32}X_k + q_{33}Y_k + q_{34}Z_k \quad (116)$$
$$Z_{k+1} = p_z + q_{42}X_k + q_{43}Y_k + q_{44}Z_k \quad (117)$$

and the output is simply

$$x_k = X_k/Z_k \quad (118)$$
$$y_k = Y_k/Z_k. \quad (119)$$

The q_{ij} in the above equations are the elements of the matrix in (29) introduced for the quaternion representation of rotations. We see that we are trading off higher accuracy for complexity of the system.

6.2.1 SSD-based displacement estimation

Displacement estimation is related to optical flow estimation in that displacements are the product of the instantaneous velocity and the inverse frame rate. While optical flow estimation has proceeded along the line suggested by Horn and Schunck [27], displacement estimation using correlation-based methods or sums-of-squared-differences has proven to be effective. Comparisons of typical correlation-based matchers can be found in Hannah [17] and Burt, Yen and Xu [9]. An in-depth study of the applications of these techniques to the estimation of displacement fields in motion sequences is presented by Anandan [2]. In particular, correlation-based estimation has proven to be useful when large interframe displacements occur. On the other hand, the method encounters difficulties with foreshortening.

We employ the sum-of-squared-differences (SSD) technique which we briefly describe below and in figure (12). Suppose we wish to determine where the image of a point in the real world that was located at coordinates (x, y) in frame t has moved to in frame $t + T$. Let the new coordinates be $(x', y') = (x + \Delta x, y + \Delta y)$. We assume that the brightness in a neighborhood of the point of interest does not change significantly from one image to the next (refer to the brightness change constraint assumption). To obtain a measure of the quality of a given displacement estimate $(\Delta x, \Delta y)$ we can therefore sum the squares of the differences in the brightness values of corresponding points in the neighborhoods of the original and displaced image point. More formally if $E(x, y, t)$ denotes the image brightness at location (x, y) in frame ' we have

$$\Theta(\Delta x, \Delta y) = \sum_{x,y\epsilon P} [E(x, y, t) - E(x + \Delta x, y + \Delta y, t + T)]^2 \quad (120)$$

as a measure Θ for the error in a displacement estimate $(\Delta x, \Delta y)$. P is used as a symbol for a set of image points constituting the neighborhood to be considered in the correlation. The displacement estimator minimizes the above error to produce $(\Delta x*, \Delta y*)$ such that

$$\Theta(\Delta x*, \Delta y*) = \min_{\Delta x, \Delta y \epsilon D} \Theta(\Delta x, \Delta y). \quad (121)$$

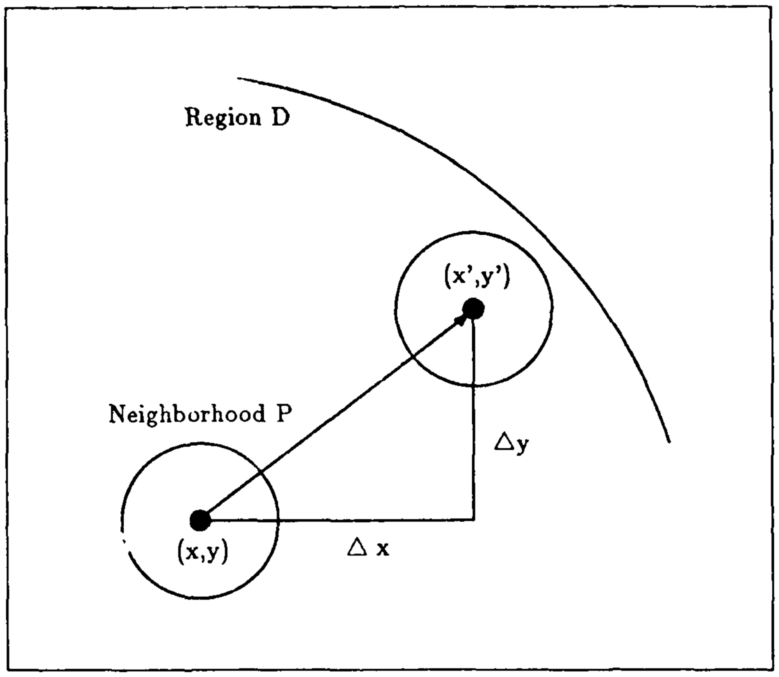


Figure 12: SSD displacement estimation

In a practical implementation such as the one by Little [33] which we intend to use in our scheme, $\Theta(\Delta x, \Delta y)$ is computed for some set of displacements D in frame $t+T$. The resulting set of error values is searched for a minimum to yield $(\Delta x*, \Delta y*)$.

In order to utilize such an estimate in a Kalman filter we need a quantification of its variance or more precisely the covariance matrix for $[x', y']^T$ which is the same as for $[\Delta x, \Delta y]^T$:

$$\tilde{\mathbf{R}} = \begin{bmatrix} \sigma^2_{\Delta x} & cov(\Delta x, \Delta y) \\ cov(\Delta x, \Delta y) & \sigma^2_{\Delta y} \end{bmatrix}. \tag{122}$$

We may assume that estimation error in the x- and y-directions are independent so that $cov(\Delta x, \Delta y) = 0$. The remaining variances are due to the variance in the brightness measurement as described in the previous section. However, the measurement procedure itself contributes to the uncertainty.

Anandan [2] suggested a "confidence measure" which was justified intuitively in the following way. The residual error $\Theta(\Delta x*, \Delta y*)$ is one component of the confidence measure: the higher it is the lower our confidence in the measure. The second component is due to the fact that there must be significant variation of brightness within the neighborhood P in order for the correlation scheme to be able to identify the neighborhood in the next image. In other words points in areas of uniform brightness are impossible to localize in successive images. The notion of variation of brightness throughout the patch is captured by the curvature of the SSD function i.e. its second derivative. The confidence measure was therefore essentially the second derivative of Θ divided by the residual error in the optimum $\Theta(\Delta x*, \Delta y*)$.

Unfortunately, there is no formal concept of a confidence measure and hence no derivation to justify it exists - it is merely intuitive. It is also intuitive that a variance should be the "inverse" of such a confidence measure although the connection to probabilistic measures is not clear either. We have found a formal derivation $\sigma^2_{\Delta x}$ and $\sigma^2_{\Delta y}$ which verify Anandan's conjecture. The derivation is rather lengthy and is given in appendix C. The derivation results in the following values for the variances

$$\sigma^2_{\Delta x} = 8\sigma^2_E \frac{(\frac{\Theta^2_{yy}}{d^2_x} - \frac{\Theta^2_{xy}}{d^2_y})\Theta}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}} \tag{123}$$

$$\sigma^2_{\Delta y} = 8\sigma^2_E \frac{(\frac{\Theta^2_{xx}}{d^2_y} - \frac{\Theta^2_{xy}}{d^2_x})\Theta}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}}. \tag{124}$$

Notice that the denominators contain the Hessian of Θ which indicates the curvature while the numerators contain the residual error Θ. These are the components of the confidence measure suggested b Anandan. The above variances are needed in the Kalman Filter update process described below.

6.2.2 Kalman Filter Update

Before the actual update can take place we must first compute the filter gain. From (21) we have

$$\mathbf{E}_k = \mathbf{P}^-_k \mathbf{C}_k [\mathbf{C}^T_k \mathbf{P}^-_k \mathbf{C}_k + \tilde{\mathbf{R}}_k]^{-1} \tag{125}$$

where

$$\mathbf{C}_k = \begin{bmatrix} \frac{1}{\hat{Z}_k} & 0 & -\frac{\hat{X}_k}{\hat{Z}_k} \\ 0 & \frac{1}{\hat{Z}_k} & -\frac{\hat{Y}_k}{\hat{Z}_k} \end{bmatrix} \tag{126}$$

is the Jacobian of the output equation and

$$\tilde{\mathbf{R}}_k = \begin{bmatrix} \sigma^2_{\Delta x} & 0 \\ 0 & \sigma^2_{\Delta y} \end{bmatrix} \tag{127}$$

is the measurement covariance computed in the previous subsection. We see the increased computational cost here as we must invert a two-by-two matrix for every pixel whereas the previous scheme involved only a scalar inverse.

Now the filter update proceeds as usual

$$\hat{\mathbf{R}}^+_k = \hat{\mathbf{R}}^-_k + \mathbf{E}_k(\begin{bmatrix} x' \\ y' \end{bmatrix} - \begin{bmatrix} \frac{\hat{X}^-_k}{\hat{Z}^-_k} \\ \frac{\hat{Y}^-_k}{\hat{Z}^-_k} \end{bmatrix}) \tag{128}$$

$$\mathbf{P}^+_k = (\mathbf{I} - \mathbf{E}_k \mathbf{C}_k)\mathbf{P}^-_k \tag{129}$$

where x' and y' are the outputs of the displacement estimator. Also note that due to the fact that we are now measuring displacements (i.e. in a sense the first two components of the state vector) rather than the depth, we must perform the depth map interpolation after the update instead of after the prediction. The predicted output is compared to the displacement estimate in the update equation above. Therefore it is not necessary that the depth values are valid at grid-points after the prediction stage. We will, however, perform the grid-point interpolation after the update to enable motion estimation and prediction to operate in the usual fashion. The interpolation proceeds as described in subsection 6.1.4.

6.2.3 Motion Estimation

In this module we recover the motion that best accomplishes the transformation between frames $k-1$ and k or rather between our estimates of the state variables at these times. If we denote the image region under consideration by P then we seek the motion parameters p_x, p_y, p_z and q_{ij} for $i, j = 2,3,4$ such that

$$\begin{aligned}\textstyle\sum_P [\quad & (X_k - p_x + q_{22}X_{k-1} + q_{23}Y_{k-1} + q_{24}Z_{k-1})^2 \quad + \\ & (Y_k - p_y + q_{32}X_{k-1} + q_{33}Y_{k-1} + q_{34}Z_{k-1})^2 \quad + \\ & (Z_k - p_z + q_{42}X_{k-1} + q_{43}Y_{k-1} + q_{44}Y_{k-1})^2 \quad]\end{aligned} \tag{130}$$

is minimized.

The necessary conditions for a minimum are the result of differentiating the above expression with respect to all of the motion parameters and equating to zero. This yields 12 linear equations for the parameters. But since the above error sum contains three independent terms, we really have 3 four-by-four systems to solve. They have the form

$$\begin{bmatrix} \sum_P 1 & \sum_P X_{k-1} & \sum_P Y_{k-1} & \sum_P Z_{k-1} \\ \sum_P X_{k-1} & \sum_P X_{k-1}^2 & \sum_P X_{k-1}Y_{k-1} & \sum_P X_{k-1}Z_{k-1} \\ \sum_P Y_{k-1} & \sum_P X_{k-1}Y_{k-1} & \sum_P Y_{k-1}^2 & \sum_P Y_{k-1}Z_{k-1} \\ \sum_P Z_{k-1} & \sum_P X_{k-1}Z_{k-1} & \sum_P Y_{k-1}Z_{k-1} & \sum_P Z_{k-1}^2 \end{bmatrix} \begin{bmatrix} p_i \\ q_{i+1,2} \\ q_{i+1,3} \\ q_{i+1,4} \end{bmatrix} = \begin{bmatrix} \sum_P R_{i,k} \\ \sum_P X_{k-1}R_{i,k} \\ \sum_P Y_{k-1}R_{i,k} \\ \sum_P Z_{k-1}R_{i,k} \end{bmatrix} \tag{131}$$

for $i = 1,2,3$. We have used the notation p_i to denote the components of the translation quaternion and $R_{i,k}$ to denote the components of $\mathbf{R}_k$, i.e. X_k, Y_k and Z_k. This somewhat alleviates the computational burden.

To complete the motion estimate, we must ensure that the matrix

$$\mathbf{Q} = \begin{bmatrix} q_{22} & q_{23} & q_{24} \\ q_{32} & q_{33} & q_{34} \\ q_{42} & q_{43} & q_{44} \end{bmatrix} \tag{132}$$

is orthonormal or in other words that the rotation quaternion has unit magnitude. We seek the nearest orthonormal matrix to $\mathbf{Q}$ which is

$$\hat{\mathbf{Q}} = \mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1/2} = \mathbf{Q}(\sqrt{\lambda_1}\mathbf{u}_1\mathbf{u}_1^T + \sqrt{\lambda_2}\mathbf{u}_2\mathbf{u}_2^T + \sqrt{\lambda_3}\mathbf{u}_3\mathbf{u}_3^T) \tag{133}$$

where λ_i denotes the ith eigenvalue of $\mathbf{Q}$ and $\mathbf{u}_i$ the corresponding unit eigenvector. This again involves quite an amount of computation.

6.2.4 Kalman Filter Prediction

We note that the Jacobian of our dynamical system Φ_k is simply the matrix $\mathbf{Q}_k$ from the previous subsection. The prediction equations become

$$\hat{\mathbf{R}}_{k+1}^{-} = \mathbf{p}_k + \mathbf{Q}_k \hat{\mathbf{R}}_k^{+} \quad (134)$$
$$\mathbf{P}_{k+1}^{-} = \mathbf{Q}_k \mathbf{P}_k^{+} \mathbf{Q}_k^T \quad (135)$$

where we have once again assumed that the plant noise is zero.

In the block-diagram we have fed the state estimate back into the displacement estimator. The idea is to use this estimate to limit the search the SSD-matcher must do. In the description of the displacement estimator we denoted the region of displacements the estimator would consider by D. This region will now be a neighborhood of the predicted displacement $(\hat{X}_k^- / \hat{Z}_k^-, \hat{Y}_k^- / \hat{Z}_k^-)$ which limits the amount of search considerably.

6.2.5 Discussion

This scheme is well suited for alleviating the problems of the previous one namely to decouple motion and depth estimation. The displacement estimator no longer requires the motion estimate for its operation. In addition our measurement process is no longer based on gradient approximations but rather draws information from a region of the image plane. We expect this to result in greater robustness and a higher rate of convergence.

The key features of the approach presented above are summarized here:

(1) An SSD-estimator is used to measure displacements.

(2) A discrete object motion model using quaternions is employed.

(3) An Extended Kalman Filter is used to estimate real-world object coordinates.

(4) The SSD-estimator is based on a brightness constancy assumption. No additional assumptions about surface, reflectance or motion are made.

(5) A dense depth map and discrete motion parameters are recovered.

(4) The scheme is computationally quite intensive. Among the time consuming operations are: searching for the optimal displacement, matrix inversions for the gain computation, solving three 4x4 systems for motion estimation, renormalizing the rotation matrix.

The high computational cost may be the main deficieny of this scheme. We intend to use the Connection Machine for the implementation of the algorithm in order to achieve near-real-time performance.

It is worth pointing out some essential differences between the algorithm we have suggested in this section and the work of Matthies, Szeliski and Kanade [37]. In both cases a Kalman Filter is used to predict and update a dense depth and variance map. However, the latter approach is restricted to translational motion parallel to the image plane. In addition, motion must be known at every point in time. We have imposed no restriction on motion and have further shown how motion estimation can be incorporated into the iterative filtering process. This makes the our approach applicable to a far more general class of imaging situations.

7 Conclusion

We have seen three conceptually different schemes for modelling the imaging situation as a dynamical system and using a state space observer/filter to recover parameters of interest such as depth and motion. The second algorithm presented differs structurally from the other two in that it recovers surface parameters instead of a dense depth map. Of course this requires a parametrization of the surface which is usually unable to handle discontinuities. But the parameter-based approach may have higher robustness and be well suited for specific applications. One such application is the landing approach of an airplane in which we may approximate the runway area as a plane and seek to recover the relative orientation of aircraft and runway. The parameter-based approach can naturally be extended to more complex surface structures such as quadratic patches and more sophisticated motion models such as constant acceleration. Another interesting idea is to approximate locally a complex surface by planar patches and apply the ideas to those patches. But before elaborating either of the models, they should prove their capabilities in an implementation.

The first algorithm presented in this paper attempted to model the correct physical relationships involved in the formation of brightness values. We saw that this forced us to make very strong assumptions about the reflectance properties of the surface and the illumination conditions. Most real images will not satisfy these conditions and the algorithm is expected to operate accurately only on synthetic data.

The two variants of integrated motion and depth-estimation presented last are most promising for application to real images. They show how the Kalman filter can be incorporated into existing motion vision schemes to achieve incremental depth estimation over more than 2 frames. We also noted that a tradeoff between accuracy/robustness and computation expense is involved of which the presented alternative implementations are good examples. The Kalman filter proved useful for the depth estimation process because this quantity could be interpreted as the state variable of a dynamical system. Motion estimation, however, is not possible without some dynamics model of the actuating device. An interesting idea would be to incorporate the dynamical model of a mobile robot, which carries the camera into the plant equations, and apply the Kalman filter for simultaneous depth and motion estimation.

The depth map approaches also reveal why the application of filtering theory has previously focused on feature matching: displacements of features are known rather precisely and provide a good measurement for the filter. On the other hand they have the deficiencies discussed earlier which led us to consider non-feature based algorithms.

It is important to mention that the dynamical systems approach is not limited to the application of filters for depth and motion estimation. There are two other applications that come to mind. What for instance would it mean to close the *feedback* loop over one of the dynamical systems presented above? It means that we measure some indication of motion in the image plane and feed it back to control the motion of the camera. This could be used in tracking or some other orientation procedure of the camera carrier such as in mobile robots or aircrafts and satellites.

Another powerful technique that one could import from control theory is *system identification.* Sophisticated techniques have been developed to estimate the *parameters* of dynamical systems which are the constants in the plant and output equations. What does it mean to perform system identification on one of our models? The model in section 4 which is based on brightness values contains the diameter of the camera lens and all models implicitly contain the focal length. We can use one of these identification procedures to measure these quantities. Identification may also aid in dealing with more complex motion models as the ones presented in equation (72). In this case we would have to perform online identification i.e. identify the parameters while the observer estimation is being performed. This may lead to interesting and powerful models.

The main objective in this paper was to introduce a systematic way of dealing with a series of frames in motion vision. We have shown that dynamical systems provide a way for dealing explicitly with time dependency and have formulated the solution to a common motion vision problem in terms of such systems. It is conceivable that is approach can be extended to other problems in which information is acquired over a series of frames such as multiframe edge-detection, segmentation, color, texture etc. Unlike previous approaches mentioned in the introduction we have attempted to found our models rigorously on physical facts only. We intend to carry out experiments to corroborate our theoretical results.

Acknowledgments

Berthold Horn encouraged me to look at the problem of filter application for the moving plane and using the BCCE as a measurement equation. Tommy Poggio, Hans-Hellmut Nagel and Ellen Hildreth made valuable comments on earlier versions of the paper, provided me with excellent references on the subject and ensured that the applicability of the algorithms to real images remained the most important concern. I would also like to thank my friends Sundar Narasimhan and Brian Subirana for the inspiring conversations we had about some of the problems in this paper.

Appendix

A System Matrices for the BCCE Observer

In section 5.3 we found that the matrix

$$\mathbf{A}_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k} \tag{136}$$

and the vector

$$\mathbf{c}_k = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k, \mathbf{y}_k} \tag{137}$$

are necessary to compute the observer gain vector $\mathbf{e}_k$. The vector function $\mathbf{f}(\mathbf{x}_k)$ is given by (74), the implicit output $g(\mathbf{x}_k, \mathbf{y}_k)$ by (75). In this appendix we give the elements of the above matrices for convenient computation. We omit the hat representing estimated values and the index k of time. We use the following abbreviations:

- The surface normal $\mathbf{n} = [n_x, n_y, n_z]^T$
- The translational displacement $\mathbf{p} = [p_x, p_y, p_z]^T$
- The position vector of an image plane point $\mathbf{r} = [x, y, 1]^T$
- The brightness gradients E_x, E_y and E_t
- A vector $\mathbf{x}$ has a corresponding purely imaginary quaternon $\overset{\circ}{x}$.
- The rotation $\overset{\circ}{q}$ is a unit quaternion so $q_0^2 = 1 - q_x^2 - q_y^2 - q_z^2$

The elements of $\mathbf{A}$ are:

$$\begin{aligned}
a_{11} &= q_0^2 + q_x^2 - q_y^2 - q_z^2 + \frac{1 + n_y p_y + n_z p_z}{(1 + \mathbf{n} \cdot \mathbf{p})^2} \\
a_{12} &= 2(q_x q_y - q_0 q_z) - \frac{n_x p_y}{(1 + \mathbf{n} \cdot \mathbf{p})^2} \\
a_{13} &= 2(q_x q_z + q_0 q_y) - \frac{n_x p_y}{(1 + \mathbf{n} \cdot \mathbf{p})^2} \\
a_{14} &= -\frac{n_x^2}{(1 + \mathbf{n} \cdot \mathbf{p})^2} \\
a_{15} &= -\frac{n_x n_y}{(1 + \mathbf{n} \cdot \mathbf{p})^2} \\
a_{16} &= -\frac{n_x n_z}{(1 + \mathbf{n} \cdot \mathbf{p})^2} \\
a_{17} &= 2n_y(q_y + \frac{q_x q_z}{q_0}) + 2n_z(q_z - \frac{q_x q_z}{q_0})
\end{aligned}$$

$$
\begin{aligned}
a_{18} &= -4n_xq_y + 2n_y(q_x + \frac{q_yq_z}{q_0}) + 2n_z(q_0 - \frac{q_y^2}{q_0}) \\
a_{19} &= -4n_xq_z + 2n_y(-q_0 + \frac{q_z^2}{q_0}) + 2n_z(q_x - \frac{q_yq_z}{q_0}) \\
\\
a_{21} &= 2(q_xq_y + q_0q_z) - \frac{n_yp_x}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{22} &= q_0^2 - q_x^2 + q_y^2 - q_z^2 + \frac{1 + n_xp_x + n_zp_z}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{23} &= 2(q_yq_z - q_0q_x) - \frac{n_yp_z}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{24} &= -\frac{n_xn_y}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{25} &= -\frac{n_y^2}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{26} &= -\frac{n_yn_z}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{27} &= 2n_x(q_y - \frac{q_xq_z}{q_0}) - 4n_yq_x + 2n_z(-q_0 + \frac{q_x^2}{q_0}) \\
a_{28} &= 2n_x(q_x - \frac{q_yq_z}{q_0}) + 2n_z(q_z + \frac{q_xq_y}{q_0}) \\
a_{29} &= 2n_x(q_0 - \frac{q_z^2}{q_0}) - 4n_yq_z + 2n_z(q_y - \frac{q_xq_y}{q_0}) \\
\\
a_{31} &= 2(q_xq_z - q_0q_y) - \frac{n_zp_x}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{32} &= 2(q_yq_z + q_0q_x) - \frac{n_zp_y}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{33} &= q_0^2 - q_x^2 - q_y^2 + q_z^2 + \frac{1 + n_xp_x + n_yp_y}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{34} &= -\frac{n_xn_z}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{35} &= -\frac{n_yn_z}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{36} &= -\frac{n_z^2}{(1+\mathbf{n}\cdot\mathbf{p})^2} \\
a_{37} &= 2n_x(q_z + \frac{q_yq_z}{q_0}) + 2n_y(q_0 - \frac{q_x^2}{q_0}) - 4n_zq_x \\
a_{38} &= 2n_x(-q_0 + \frac{q_y^2}{q_0}) + 2n_y(q_z - \frac{q_xq_y}{q_0}) - 4n_zq_y \\
a_{39} &= 2n_x(q_x + \frac{q_yq_z}{q_0}) + 2n_y(q_y - \frac{q_xq_z}{q_0})
\end{aligned} \tag{138}
$$

$$a_{ij} = 0 \qquad \text{for } i = 4, \ldots, 9 \quad j = 1, \ldots, 9 \tag{139}$$

The elements of **c** are:

$$\begin{aligned}
c_1 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](\mathbf{s} \cdot \mathbf{p})x \\
c_2 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](\mathbf{s} \cdot \mathbf{p})y \\
c_3 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](\mathbf{s} \cdot \mathbf{p}) \\
c_4 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](\mathbf{n} \cdot \mathbf{r})E_x \\
c_5 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](\mathbf{n} \cdot \mathbf{r})E_y \qquad (140) \\
c_6 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](\mathbf{n} \cdot \mathbf{r})(E_t - xE_x - yE_y) \\
c_7 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](2yE_y(q_y + \frac{q_x q_z}{q_0}) + 2E_t(q_z - \frac{q_x q_y}{q_0})) \\
c_8 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](2xE_x(q_x - \frac{q_y q_z}{q_0}) + 2E_t(q_z + \frac{q_x q_y}{q_0})) \\
c_9 &= 2 \sum_{x=-w}^{w} \sum_{y=-h}^{h} [(\mathring{q}\mathring{r}\mathring{q}^* + \mathring{p}(\mathring{n} \cdot \mathring{r})](2xE_x(q_x + \frac{q_y q_z}{q_0}) + 2xE_x(q_y - \frac{q_x q_z}{q_0}))
\end{aligned}$$

B An Efficient Pixel Interpolation Scheme

When computing Kalman filter updates or predictions of depth and variance maps the problem arises that the updated value is no longer valid at a grid-point of the pixel-array because the projection of the point under observation has moved. The proposed solution in sections 6.1.4 and 6.2.2 was to reinterpolate the grid-point values.

Here, we give a computationally efficient solution for the following problem: Given n points (x_i, y_i) and a function value $Z(x_i, y_i)$ at those points interpolate Z at a point (x, y).

We will compute the interpolated function value as a weighted sum

$$Z(x', y') = \sum_{i=1}^{n} w_i Z(x_i, y_i). \tag{141}$$

A good weighting function should fulfill the following requirements:

- $0 \leq w_i \leq 1$

- $\sum_{i=1}^{n} w_i = 1$
- The w_i should decrease as the distance $d_i^2 = (x_i - x')^2 + (y_i - y')^2$ decreases.

We therefore choose the weighting factors to be

$$w_i = \frac{\frac{1}{d_i^2}}{\sum_{i=1}^{n} \frac{1}{d_i^2}}. \tag{142}$$

This clearly fulfills all of the requirements but some special cases must be considered. Suppose all n estimates involved in the interpolation have equal distance $d_i = d$ from the grid-point. In this case

$$w_i = \frac{\frac{1}{d^2}}{\sum_{i=1}^{n} \frac{1}{d^2}} = \frac{1}{n} \tag{143}$$

which means that all estimates are weighted equally as one would expect. A more tricky case occurs when some number k of the estimates are actually located at the grid-point, i.e. $d_i = 0$ for $i = 1, \ldots, k$. For estimates on the grid-point we can rearrange the expression for the weights (142) to obtain

$$w_i = \frac{1}{\frac{d_k^2}{d_1^2} + \cdots \frac{d_k^2}{d_k^2} + \sum_{i=k+1}^{n} \frac{d_k^2}{d_i^2}} \rightarrow \frac{1}{k} \tag{144}$$

as $d_i \rightarrow 0$ for $i = 1, \ldots, k$. Similarly we rearrange the expression for the weights in the case of an estimate that does not coincide with the grid-point

$$w_j = \frac{\frac{d_k^2}{d_j^2}}{\frac{d_k^2}{d_1^2} + \cdots \frac{d_k^2}{d_k^2} + \sum_{i=k+1}^{n} \frac{d_k^2}{d_i^2}} \rightarrow 0 \tag{145}$$

as $d_i \rightarrow 0$ for $i = 1, \ldots, k$. In other words we ignore all estimates that are not on the grid-point and obtain the interpolated value as the mean of the estimates located at the grid-point.

This may be very easily encoded into an efficient $\Theta(n)$ algorithm which we give in the following pseudocode notation:

```
$k \leftarrow 0; W \leftarrow 0;$
For $i \leftarrow 1, \ldots, n$
    $d_i^2 \leftarrow (x_i - x')^2 + (y_i - y')^2;$
    if $d_i^2 = 0$ then $k \leftarrow k + 1$ else $W \leftarrow W + \frac{1}{d_i^2};$
if $k > 0$ then
    For $i \leftarrow 1, \ldots, n$
        if $d_i^2 = 0$ then $w_i \leftarrow \frac{1}{k}$ else $w_i \leftarrow 0$;
else
    For $i \leftarrow 1, \ldots, n$
        $w_i \leftarrow \frac{1}{d_i^2}/W$;
```

When dealing with real estimates they will rarely fall directly onto the grid-point and we may wish to replace the test $d_i^2 = 0$ in the above algorithm with $d_i^2 < \epsilon$ where ϵ is chosen according to the numerical accuracy of our processor.

C Variance of the SSD-Displacement Estimator

In this appendix we give the derivation of the variances $\sigma^2_{\Delta x}$ and $\sigma^2_{\Delta y}$ of an SSD displacement estimate. Recall that in section 6.2.1 we have introduced the SSD functional 120

$$\Theta(\Delta x, \Delta y) = \sum_{x,y \epsilon P} [E(x,y,t) - E(x + \Delta x, y + \Delta y, t + T)]^2 \tag{146}$$

which quantified the quality of a match between point (x, y) in frame t and point $(x + \Delta x, y + \Delta y)$ in frame $t + T$. The optimal displacement $(\Delta x*, \Delta y*)$ was defined by

$$\Theta(\Delta x*, \Delta y*) = \min_{\Delta x, \Delta y \epsilon D} \Theta(\Delta x, \Delta y). \tag{147}$$

In the following we abbreviate the second partial derivatives of Θ with respect to Δx and Δy by Θ_x and Θ_y and the second partials by Θ_{xx}, Θ_{xy} and Θ_{yy}.

The displacement estimate is the result of minimizing (146). Differentiating this equation with respect to Δx and Δy yields the necessary conditions for the minimum:

$$\Theta_x(\Delta x, \Delta y) = -2 \sum_{x,y \epsilon P} [E(x,y,t) - E(x + \Delta x, y + \Delta y, t + T)] \left. \frac{\partial E}{\partial x} \right|_{(x+\Delta x, y+\Delta y, t+T)} \tag{148}$$

$$\Theta_y(\Delta x, \Delta y) = -2 \sum_{x,y \epsilon P} [E(x,y,t) - E(x + \Delta x, y + \Delta y, t + T)] \left. \frac{\partial E}{\partial y} \right|_{(x+\Delta x, y+\Delta y, t+T)} \tag{149}$$

must both be equal to 0. Assuming that the variance in a measurement of E is σ_E^2 and in a gradient measurement of $\frac{\partial E}{\partial x}$ is $2\sigma_E^2/d_x^2$ (by applying the variance-propagation (87) to the gradient approximations (90) above) we can determine the variance in Θ_x and Θ_y.

We first introduce two new sets of random variables $A_x(x, y, t)$ and $A_y(x, y, t)$ where

$$A_x(x,y,t) = [E(x,y,t) - E(x + \Delta x, y + \Delta y, t + T)] \left. \frac{\partial E}{\partial x} \right|_{(x+\Delta x, y+\Delta y, t+T)} \tag{150}$$

$$A_y(x,y,t) = [E(x,y,t) - E(x + \Delta x, y + \Delta y, t + T)] \left. \frac{\partial E}{\partial y} \right|_{(x+\Delta x, y+\Delta y, t+T)} \tag{151}$$

which are simply the terms being summed in (148) and (149). Using the formula for the propagation of variances of independent random variables (87) we easily see that

$$\begin{aligned}\sigma^2_{A_x} &= 2\sigma^2_E[(\frac{\partial E}{\partial x}\Big|_{(x+\Delta x,y+\Delta y,t+T)})^2 + \\ &\quad \frac{1}{d_x^2}(E(x,y,t) - E(x+\Delta x, y+\Delta y, t+T))^2] \qquad (152)\\ \sigma^2_{A_y} &= 2\sigma^2_E[(\frac{\partial E}{\partial y}\Big|_{(x+\Delta x,y+\Delta y,t+T)})^2 + \\ &\quad \frac{1}{d_y^2}(E(x,y,t) - E(x+\Delta x, y+\Delta y, t+T))^2]. \qquad (153)\end{aligned}$$

With these abbreviations we can write

$$\begin{aligned}\Theta_x &= -2\sum_{x,y\epsilon P} A_x(x,y,t) \qquad (154)\\ \Theta_y &= -2\sum_{x,y\epsilon P} A_y(x,y,t). \qquad (155)\end{aligned}$$

Again we make use of the variance propagation formula (87) to obtain

$$\begin{aligned}\sigma^2_{\Theta_x} = 4\sum_{x,y\epsilon P}\sigma^2_{A_x} &= 8\sigma^2_E(\sum_{x,y\epsilon P}(\frac{\partial E}{\partial x}\Big|_{(x+\Delta x,y+\Delta y,t+T)})^2 + \\ &\quad \frac{1}{d_x^2}\sum_{x,y\epsilon P}(E(x,y,t) - E(x+\Delta x, y+\Delta y, t+T))^2) \qquad (156)\\ \sigma^2_{\Theta_y} = 4\sum_{x,y\epsilon P}\sigma^2_{A_y} &= 8\sigma^2_E(\sum_{x,y\epsilon P}(\frac{\partial E}{\partial y}\Big|_{(x+\Delta x,y+\Delta y,t+T)})^2 + \\ &\quad \frac{1}{d_y^2}\sum_{x,y\epsilon P}(E(x,y,t) - E(x+\Delta x, y+\Delta y, t+T))^2) \qquad (157)\end{aligned}$$

where we recognize the second sum in each of the expressions to be Θ.

On the other hand we can use the variance-propagation (87) on (146) to determine the immediate dependency

$$\begin{aligned}\sigma^2_{\Theta_x} &= \Theta^2_{xx}\sigma^2_{\Delta x} + \Theta^2_{xy}\sigma^2_{\Delta y} \qquad (158)\\ \sigma^2_{\Theta_y} &= \Theta^2_{xy}\sigma^2_{\Delta x} + \Theta^2_{yy}\sigma^2_{\Delta y}. \qquad (159)\end{aligned}$$

It remains to substitute the expressions for $\sigma^2_{\Theta_x}$ and $\sigma^2_{\Theta_y}$ from (157) into the above equations and solve for the desired variances $\sigma^2_{\Delta x}$ and $\sigma^2_{\Delta y}$ of the displacement components.

$$\begin{aligned}\sigma^2_{\Delta x} = \frac{\Theta^2_{yy}\sigma^2_{\Theta_x} - \Theta^2_{xy}\sigma^2_{\Theta_y}}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}} &= \frac{8\sigma^2_E}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}}[\Theta^2_{yy}(\frac{\Theta}{d_x^2} + \sum_{x,y\epsilon P}(\frac{\partial E}{\partial x}\Big|_{(x+\Delta x,y+\Delta y,t+T)})^2) - \\ &\quad \Theta^2_{xy}(\frac{\Theta}{d_y^2} + \sum_{x,y\epsilon P}(\frac{\partial E}{\partial y}\Big|_{(x+\Delta x,y+\Delta y,t+T)})^2)] \qquad (160)\end{aligned}$$

$$\sigma^2_{\Delta y} = \frac{\Theta^2_{xx}\sigma^2_{\Theta_y} - \Theta^2_{xy}\sigma^2_{\Theta_x}}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}} = \frac{8\sigma^2_E}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}}[\Theta^2_{xx}(\frac{\Theta}{d^2_y} + \sum_{x,y\epsilon P}(\frac{\partial E}{\partial y}\Big|_{(x+\Delta x, y+\Delta y, t+T)})^2) - \Theta^2_{xy}(\frac{\Theta}{d^2_x} + \sum_{x,y\epsilon P}(\frac{\partial E}{\partial x}\Big|_{(x+\Delta x, y+\Delta y, t+T)})^2)] \tag{161}$$

We can obtain an expression in analogy to the confidence measure suggested by Anandan by observing that the pixel distances d_x and d_y are small quantities so that the parts of the numerators which have a coefficient $1/d_x^2$ or $1/d_y^2$ will tend to dominate the remaining terms. The expressions for the variance are then

$$\sigma^2_{\Delta x} = 8\sigma^2_E \frac{(\frac{\Theta^2_{yy}}{d^2_x} - \frac{\Theta^2_{xy}}{d^2_y})\Theta}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}} \tag{162}$$

$$\sigma^2_{\Delta y} = 8\sigma^2_E \frac{(\frac{\Theta^2_{xx}}{d^2_y} - \frac{\Theta^2_{xy}}{d^2_x})\Theta}{\Theta^2_{xx}\Theta^2_{yy} - \Theta^4_{xy}} \tag{163}$$

in which we recognize the denominators to contain the Hessian of Θ which indicates the curvature. All expressions containing Θ must of course be evaluated for the estimated displacement $(\Delta x*, \Delta y*)$. In practice it may be too tedious to approximate all the necessary derivatives of Θ when computing the variances so that further simplifications are necessary.

References

[1] I. E. Abdou and K. Y. Wong. *Analysis of Linear Interpolation Schemes for Bi-Level Image Applications.* Journal of Research and Development 6, IBM, November 1982.

[2] P. Anandan. *Computing Dense Displacement Fields with Confidence Measures in Scenes Containing Occlusion.* COINS Technical Report 84-32, University of Massachusetts, Amherst, December 1984.

[3] A. Bandopadhay, B. Chandra, and D. H. Ballard. Active navigation: tracking and environmental point considered beneficial. In *Proceedings Workshop on Motion: Representation and Analysis*, May 1986.

[4] R. Bernstein. *Digital Image Processing of Earth Observation Data.* Journal of Research and Development, IBM, January 1976.

[5] S. Bharwani, E. Riseman, and A. Hanson. Refinement of environment depth maps over multiple frames. In *Proceedings of the Workshop on Motion: Representation and Analysis*, Charleston, S. C., May 1986.

[6] T. J. Broida and R. Chellappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1), January 1986.

[7] T. J. Broida and R. Chellappa. Kinematics and structure of a rigid object from a sequence of noisy images. In *Proceedings of the IEEE Workshop on Motion: Representation and Analysis*, Charleston, SC, May 1986.

[8] T. J. Broida and R. Chellappa. Kinematics of a rigid object from a sequence of noisy images: a batch approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Miami Beach, FL, June 1986.

[9] P. J. Burt, C. Yen, and X. Xu. Local correlation measures for motion analysis: a comparative study. In *Proceedings of the IEEE Conference on Pattern Recognition and Image Processing*, 1982.

[10] K. H. Cornog. *Smooth Pursuit and Fixation for Robot Vision.* Master's thesis, Massachusetts Institute of Technology, 1985.

[11] E. D. Dickmans. 4d-szenenanalyse mit integralen raum-/zeitlichen modellen. In *Mustererkennung 1987, 9. DAGM Symposium*, Braunschweig, W. Germany, September/October 1987.

[12] C. L. Fennema and W. B. Thompson. Velocity determination in scenes containing several moving objects. *Computer Graphics and Image Processing*, 9(4), April 1979.

[13] G. F. Franklin and J. D. Powell. *Digital Control of Dynamic Systems.* Addison-Wesley, Reading, MA, 1980.

[14] G. F. Franklin, J. D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems.* Addison-Wesley, Cambridge, MA, 1986.

[15] B. Friedland. *Control System Design.* McGraw-Hill, New York, NY, 1986.

[16] A. (Ed.) Gelb. *Applied Optimal Estimation.* The MIT Press, Cambridge, MA, 1974.

[17] M. J. Hannah. *Computer Matching of Areas in Stereo Images.* AI Memo 239, Stanford University, 1974.

[18] J. Heel. Mathematical framework for motion vision in continuous time. November 1987. Working Paper.

[19] E. C. Hildreth. The computation of the velocity field. *Proceedings of the Royal Society of London B*, 221, 1984.

[20] E. C. Hildreth. *The Measurement of Visual Motion.* The MIT Press, 1984.

[21] M. W. Hirsch and S. Smale. *Differential Equations, Dynamical Systems, and Linear Algebra.* Academic Press, New York, NY, 1974.

[22] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4, April 1987.

[23] B. K. P. Horn. Hillshading and the reflectance map. *Geo-Processing*, 2, 1982.

[24] B. K. P. Horn. *Relative Orientation.* AI Memo 994, MIT Artificial Intelligence Laboratory, September 1987.

[25] B. K. P. Horn. *Robot Vision.* The MIT Press, 1986.

[26] B. K. P. Horn. Understanding image intensities. *Artificial Intelligence*, 8, 1977.

[27] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17, 1981.

[28] B. K. P. Horn and E. J. Weldon, Jr. Computationally-efficient methods for recovering translational motion. In *Proceedings of the IEEE First International Conference on Computer Vision*, London, England, June 1987.

[29] B. K. P. Horn and E. J. Weldon, Jr. Robust direct methods for recovering motion. *International Journal of Computer Vision*, April 1987.

[30] T. Kailath. *Linear Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1980.

[31] K. Kanatani. Structure from motion without correspondence: general principle. In *Proceedings Image Understanding Workshop*, Miami, FL, December 1985.

[32] B. J. Kuo. *Digital Control Systems.* Holt, Rinehart and Winston, Inc., 1980.

[33] J. Little, H. Bulthoff, and T. Poggio. Parallel optical flow computation. In *DARPA Image Understanding Workshop*, Los Angeles, CA, 1987.

[34] L. Ljung. *System Identification: Theory for the User.* Prentice-Hall, Englewood Cliffs, NJ, 1987.

[35] H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In S. Ullman and W. Richards, editors, *Image Understanding 1984*, Ablex, 1984.

[36] D. G. Luenberger. *Introduction to Dynamic Systems.* John Wiley and Sons, Inc., 1979.

[37] L. Matthies, R. Szeliski, and R. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. In *Proceedings of the DARPA Image Understanding Workshop*, Cambridge, MA, April 1988.

[38] A. Mitiche. On kineopsis and computation of structure and motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8, 1986.

[39] H.-H. Nagel. Constraints for the estimation of displacement vector fields from image sequences. In *Proceedings International Joint Conference on Artificial Ingelligence*, Karlsruhe, W. Germany, 1983.

[40] H.-H. Nagel. On the estimation of dense displacement vector fields from image sequences. In *Proceedings ACM SIGGRAPH/SIGART Interdisciplinary Workshop on Motion: Representation and Perception*, Toronto, Canada, April 1983.

[41] H.-H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(5), September 1986.

[42] S. Negahdaripour and B. K. P. Horn. Direct passive navigation: analytical solution for planes. In *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 1986.

[43] J. P. Norton. *An Introduction to Identification.* Academic Press, Orlando, FL, 1986.

[44] K. Ogata. *Discrete-Time Control Systems.* Prentice-Hall, Englewood Clifs, NJ, 1987.

[45] K. Prazdny. Egomotion and relative depth map from optical flow. *Biological Cybernetics*, 36, 1980.

[46] S. S. Rifman and D. M. McKinnon. *Evaluation of Digital Correction Techniques - for ERTS Images.* Technical Report E74-10792, TRW Systems Group, July 1974.

[47] J. W. Roach and J. K. Aggarwal. Determining the movement of objects from sequences of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(6), November 1980.

[48] J. Rooney. A survey of representations of spatial rotation about a fixed point. *Environment and Planning B*, 4, 1977.

[49] I. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(1), January 87.

[50] H. Shariat and K. E. Price. How to use more than two frames to estimate motion. In *Proceedings of the Workshop on Motion: Representation and Analysis*, Charleston, S. C., May 1986.

[51] J. H. Stuelpnagle. On the parametrization of the three-dimensional rotation group. *SIAM Review*, 6(4), October 1964.

[52] J. Stuller and G. Krishnamurthy. Kalman filter formulation of low-level television motion estimation. *Computer Vision, Graphics and Image Processing*, 21(2), February 1983.

[53] R. H. Taylor. Planning and execution of straight line manipulator trajectories. In M. J. Brady, J. M. Hollerbach, T. L. Johnson, T. Lozano-Peréz, and M. T. Mason, editors, *Robot Motion: Planning and Control*, The MIT Press, Cambridge, MA, 1982.

[54] R. Y. Tsai. Multiframe image point matching and 3-d surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2), March 1983.

[55] R. Y. Tsai and S. T. Huang. Estimating three-dimensional motion parameters of a rigid planar patch. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-29(6), December 1981.

[56] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1), January 1984.

[57] B. Tuong-Phong. *Illumination for computer-generated images.* Technical Report UTEC-CSc-73-129, Computer Science Department, University of Utah, July 1973.

[58] S. Ullman. *Maximizing Rigidity: The Incremental Recovery of 3-D Structure from Rigid and Rubbery Motion.* AI Memo 721, MIT Artificial Intelligence Laboratory, June 1983.

[59] A. Verri, F. Girosi, and V. Torre. The mathematical properties of the 2d motion field: from singular points to motion parameters. January 1988. Draft.

[60] A. Verri and T. Poggio. *Motion Field and Optical Flow: Qualitative Properties.* AI Memo 917, MIT Artificial Intelligence Laboratory, December 1986.

[61] A. M. Waxman and S. S. Sinha. Dynamic stereo: passive ranging to moving objects from relative image flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(4), July 1986.

[62] A. M. Waxman and S. Ullman. Surface structure and three-dimensional motion from image flow kinematics. *International Journal of Robotics Research*, 4(3), 1985.

[63] A. M. Waxman and K. Wohn. Contour evolution, neighborhood deformation, and global image flow: planar surfaces in motion. *International Journal of Robotics Research*, 4(3), 1985.

[64] J. Weng, T. S. Huang, and N. Ahuja. 3-d motion estimation, understanding, and prediction from noisy image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(3), May 1987.

[65] A. S. Willsky. *Digital signal processing and control and estimation theory.* The MIT Press, Cambridge, MA, 1979.